Diego Tosato

Tensor Representations for Object Classification and Detection

March 23, 2012

Università degli Studi di Verona
Dipartimento di Informatica

Advisor:
Vittorio Murino
Università di Verona
Istituto Italiano di Tecnologia (IIT)

Co-Advisor:
Marco Cristani
Università di Verona
Istituto Italiano di Tecnologia (IIT)

Mauro Spera
Università di Verona

Examiner:
Fatih Porikli
Mitsubishi Electric Research Labs (MERL)

Marcello Pelillo
Università di Venezia

Series N°: **TD-10-12**

Università di Verona
Dipartimento di Informatica
Strada le Grazie 15, 37134 Verona
Italy

To my wife Stefania.

Abstract

A key problem in object recognition is finding a suitable object representation. For historical and computational reasons, vector descriptions that encode particular statistical properties of the data have been broadly applied. However, employing tensor representation can describe the interactions of multiple factors inherent to image formation. One of the most convenient uses for tensors is to represent complex objects in order to build a discriminative description.

Thus thesis has several main contributions, focusing on visual data detection (e.g. of heads or pedestrians) and classification (e.g. of head or human body orientation) in still images and on machine learning techniques to analyse tensor data. These applications are among the most studied in computer vision and are typically formulated as binary or multi-class classification problems.

The applicative context of this thesis is the video surveillance, where classification and detection tasks can be very hard, due to the scarce resolution and the noise characterising sensor data. Therefore, the main goal in that context is to design algorithms that can characterise different objects of interest, especially when immersed in a cluttered background and captured at low resolution.

In the different amount of machine learning approaches, the ensemble-of-classifiers demonstrated to reach excellent classification accuracy, good generalisation ability, and robustness of noisy data. For these reasons, some approaches in that class have been adopted as basic machine classification frameworks to build robust classifiers and detectors. Moreover, also kernel machines has been exploited for classification purposes, since they represent a natural learning framework for tensors.

Contents

List of Figures

List of Algorithms

List of Tables

1

Introduction

Contents

1.1 Structure of the Thesis

The goal of computer vision is to extract useful information from images. This has proved to be a surprisingly challenging task. Part of the problem is the complexity of visual data. Consider the image in Fig. 1.1. There are many people in this video surveillance scene. Almost none of these are presented in a typical pose. Almost all of them are partially occluded. For a computer vision algorithm, it is not even easy to establish where one person ends and another begins. However, computer vision is not impossible, but it is very challenging. Perhaps this was not appreciated at first because what one perceives when looks at a scene is already highly processed. Nonetheless, computer vision algorithms can sometimes beat the human vision system. For example, consider Fig. 1.2, where the goal is to infer the orientation of the head. Observing the smallest images, it appears quite difficult to guess the orientation for the human eye, but in this thesis a framework able to infer the orientation for those images with a surprisingly good accuracy is proposed. Another example is the case of a network of security cameras which should be monitored in order to find abnormal events. A human cannot focus his attention to many video stream provided by cameras, while a computer vision algorithm can control all the streams at the same time, having a "global" understanding of the monitored area.

At an abstract level, the goal of computer vision approach is to use the observed image data to infer something about the world. For example, one may build a method that, observing a frame of a video sequence, detects the objects contained automatically. To solve a (computer vision) problem of this type, one needs three components: (1) a model that mathematically relates the visual data $\mathbf{x}$ and the world state c (considering the proposed example, c is the category of the objects). The model specifies a family of possible relationships between $\mathbf{x}$ and

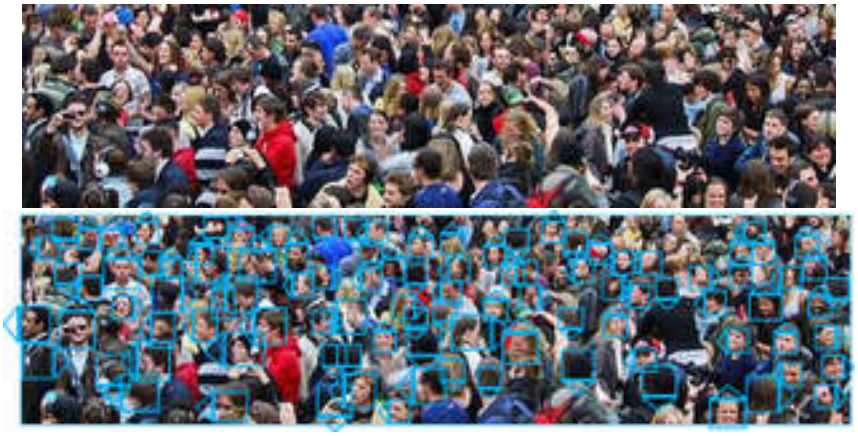


Fig. 1.1. On the top, a visual scene containing many people, almost all of which are partially occluded. On the bottom the processing of the scene where orientated boxes indicate a rough classification of the human gaze that can be used in video surveillance applications.

Fig. 1.2. The object resolution issue.

c and the particular relationship is determined by the model parameters Θ; (2) a learning algorithm that allows to fit the parameters Θ using paired training examples $\{\mathbf{x}_i, c_i\}_{i=1,\ldots,N}$, where N is the number of training examples; (3) an inference method that takes a new observation $\mathbf{x}$ and uses the model to return the posterior $P(c|\mathbf{x}, \Theta)$ over the world state c.

The first and most important component of the solution is the model [Bel06, HTF11, Pri12]. Every model relating the data $\mathbf{x}$ to the world c falls into one of two categories.

- Models that directly infer the world state on the data as $P(c|\mathbf{x})$.
- Models that indirectly infer the world state on the data modelling $P(\mathbf{x}|c)$ or $P(\mathbf{x}, c)$.

The first type of model is termed *discriminative*. The second is termed *generative* and can be used to generate new observations. It is not possible to establish which is the best type of model that should be adopted, but it depends on the problem tackled. Among the computer vision problems, in this thesis the classification and detection of defined class of objects are faced.

Ideally, when one trains a model for detection or classification, this could be generative: it must produce a probability distribution $p(\mathbf{x}|c)$ to measure the statistics for an object $\mathbf{x}$ and its class label c. Unfortunately, generative models are often out of reach or their computational burden is too high. For example, cars are a relatively easy class to study, but there is no existing generative model which captures all the variations such as, texture, multi-view, shape, etc. Alternatively,

one can look for a discriminative model $p(c|\mathbf{x})$ in which c is a simple label that identifies the object class. I am interested in the latter type of models.

However, for classification and detection purposes, there is another important aspect to be considered, that is how to represent an object. This issue concerns how the measurement vector $\mathbf{x}$ was created. In state-of-the-art vision systems, the image pixel data is almost always processed to form the measurement vector. The idea under this measurement process is the following: the image data may be contingent on many aspects of the real world that do not pertain to the task at hand. For example, in an object detection task the RGB values change depending on the camera gain, illumination, object pose and particular instance of the object (some examples are reported in Fig. 1.3). Therefore one may want to remove

Fig. 1.3. Some issues of the real world images.

as much "noise" as possible while retaining the aspects of the image that are critical to the final decision (the choice of the c label). It should be emphasized that this step, also known as *feature extraction*, is very important. In practice the choice of the right features can influence the performance of vision systems at least as much as the choice of model. For robust object classification and detection it is crucial to characterize the regions of an image in a way that is compact and stable to changes in the image. To this purpose, Lazebnik et al. [LSP06] use SIFT (Scale-Invariant Feature Transform) descriptor extracted from a regular grid; Dalal & Triggs develop the HOG descriptor [DT05]; Forssen & Lowe (2007) develop a descriptor for use with maximally stable extremal regions; local binary patterns are implemented in Ojala et al. [OPM02]; Tuzel et al. [TPM06] develop a very effective descriptor based on region covariance information. Recent works on image descriptors have applied machine learning techniques to optimize their performance in Brown et al. [BHW11] and Philbin et al. [PISZ10]).

For the detection or classification of visual objects the human vision system uses several cues. Therefore feature extraction phases which consider only a single source of information (like using the colour information, the shape information or the motion information) rarely achieve good accuracy performances if compared with the human vision system. So the feature extraction can be handled extracting and combining multiple meaningful features in one tensor (non-vector) representation. The reliability of a tensor representation depends on the robustness to operate on noisy data.

Tensor representation meets a dramatic limitation of the classical machine learning techniques that is the assumption that the object representation is a vector. However, in the recent past, a rising interest in how to deal with tensor data is clearly visible. In fact, some machine learning approaches have been proposed

for smooth manifolds (Stiefel manifolds, Grassmann manifolds, Riemannian manifolds, etc.), where tensors can be properly represented. In this thesis, I present different tensor representations for visual objects. I focus my attention on the symmetric, symmetric positive-definite tensors (or SPD tensors), and set-of-vectors of real numbers.

For computational purposes, one can directly use a flat Euclidean structure to define a metric on tensors. This is an efficient solution, but, in general, it is not always satisfactory. In fact, tensors have a specific matrix structure, that can be better managed with other more appropriate metrics. In this thesis an effective measure of the non-flatness of a set of tensors is proposed. This can be used to estimate the error occurring due to the use of the Euclidean distance.

My thesis is organized as follows. In Chapter 2, the fundamental mathematical tools used to deal with tensors are described; this because tensors have a "non-vector" form, so that the basic tools utilized to manipulate them are different from the standard vector ones.

In Chap. 3, the discriminative learning models adopted are introduced. I have striven considerably to extend some methods belonging to the discriminative class and to build different approaches to learn from tensors, for object detection and multi-class classification problems at the same time.

In Chap. 4, a new tensor descriptors of image features (like colour, gradient information, etc.) computed inside an image region is described. It is shown that these tensor representations lead to better performances compared with state-of-the-art tensor representations for classification and detection problems.

In Chap. 5 the attention is focused on the object detection (i.e. pedestrian detection) task. Here different detection architectures are proposed in order to tackle several issues related to the detection, as the efficiency, the problem of the object occlusion, and the problem of the detection of small pedestrians (which are typical in video surveillance scenarios). At the end of this Chapter, the application of some of the proposed detection frameworks is shown, applied to the data used in the SAMURAI video surveillance project [sam].

In Chap. 6, different frameworks for the classification and regression problems exploiting the tensor representation are described. In this Chapter the most important theoretical contributions regarding how to exploit tensors and their manifold structure in a theoretically sound way are contained. Moreover, an important application of one of the framework presented into a social-signalling application is described.

Finally, conclusions and possible new directions for future research are presented in Chap. 7.

1.2 Contributions of the Thesis

The thesis presents several contributions, for the classification and detection problems. It proposes different tensor representations of visual objects to characterizing their content (Sec. 4.2.1, 4.2.2, 4.2.3) and structure (Sec. 4.2.3, 4.2.4). In particular, for what regards the pedestrian detection task, four object architectures, exploring different kinds of tensors are outlined; i.e. a framework based on automatic feature selection made using Boosting (Sec. 5.2) which improves the state-of-the-art

pedestrian detector [TPM08], a part-based pedestrian detector on multiple tangent spaces (one for every part) based on covariance tensors (Sec. 5.3), a low resolution pedestrian detector based on self-similarity tensors (Sec. 5.4), and a robust to occlusion set-based pedestrian detection framework where the body configuration is not fixed (Sec, 5.5). Moreover, it proposes a new class of features referred to as ARCO (Sec. 6.3) which is further evolved to WARCO (Sec. 6.4) and FWARCO (Sec. 6.4) for the description of low resolution objects on different regression and multi-class computer vision tasks, such as head orientation classification, human orientation classification, pedestrian classification, head pose estimation. For all these tasks novel datasets are introduced (Sec. 6.4.4); they are freely available at [Tosa]. In addition, it introduces a novel criterion (Sec. 6.3.2,6.4.2), based on the Riemannian curvature, to estimate the non-flattens of a set of tensors, which can be used to estimate the error committed in approximating tensors on a Euclidean manifold for learning purposes. That criterion is valid over any connected Riemannian manifold. Besides, it describes a way to find possible approximations of the actual distance among tensors that can be combined with standard machine learning algorithms for multi-class classification and regression problems. Finally, it presents novel classification architecture for embedded computer vision devices exploiting tensors (Sec. 5.6).

2

Fundamental Mathematical Tools

Contents

2.1 Introduction

The problems involving matrix (tensor) manifold appear in a wide variety of machine learning and computer vision tasks. In this Chapter, the fundamental tools to manipulate matrices are presented. They will be used in several parts of the rest

of the thesis. For a thorough treatment the reader is referred to the related bibliography [Kel75, DC92, FK97, Bre97, DK00, Spe, AMS08, PP08, Hat02, Ber03, BBBK08, GHL04, Gal11]. To write this Section concepts from different books and lecture notes are merged, i.e. [AMS08, Spe, Ber03, PP08] adding examples of matrix manifolds utilized in this thesis and discussing the concepts as simply as possible to provide the reader with the necessary tools to understand the mathematical problems tackled in the thesis together with its mathematical contributions.

The concept of matrix manifold, and in general of manifold, is one of the most important in mathematics. To get an idea of what a manifold is, think of the surface of a sphere, or of a torus. If you cut out a very small piece of one of these surfaces, then it looks like a sheet of paper or, to be more precise, it is locally Euclidean. This means that if you consider a very small region in a surface, then its geometry is just like the ordinary two-dimensional geometry of the plane. However, the global behaviour of manifolds is not Euclidean in general.

There are two important points to make about manifolds. First, recalling the previous examples, a sphere and a torus are naturally visualized as surfaces that live inside a three-dimensional space, but it is possible to talk about manifolds intrinsically. That is, you can discuss the geometry of a manifold by focusing on the points in the manifold itself and making no reference to any external space in which the manifold lives. Secondly, there can be manifolds of any dimension. In this thesis I use matrix manifolds which are not easy to visualize, but exist abstractly. To conclude this brief explanation of the concept of manifold, I wish to make an important comment. From the previous characterisation it is obvious that manifolds are topological objects and they do not involve a priori a notion of distance between points. However, we shell deal with manifolds equipped with a metric, namely, Riemannian manifolds.

This chapter is organized as follows: Sec. 2.2 presents some fundamental operations on matrices which are necessary to understand both the theoretical part of this thesis and also the proposed approaches. Sec. 2.3 describe a minimal set of concepts to approach the manifolds' universe. In Sec. 2.4 matrix manifolds are introduced with particular attention to the quotient manifolds which are widely exploited on this thesis. Then the Riemannian geometry, Lie Groups, and symmetric spaces are introduced in Sec. 2.5. Finally, two matrix manifolds of interest are described in detail in Sec. 2.6.

2.2 Fundamental Matrix Algebra

Matrices are used extensively throughout this thesis and are written in bold (e.g., $\mathbf{X}$,$\mathbf{Y}$). I introduced some fundamental operations on matrices sampling form the excellent Matrix Cookbook reported in [PP08] and adding some notions which are specific for the SPD and symmetric matrices.

2.2.0.1 Matrices

Here, the attention is focused on square matrices (same number of columns and rows). They are always indexed by row first and then column, so x_{ij} denotes the element of matrix $\mathbf{X}$ at the i-th row and the j-th column.

X diagonal matrix is a square matrix with zeros everywhere except on the diagonal. An important special case of a diagonal matrix is the identity matrix **I**. This has zeros everywhere except for the diagonal where all the elements are equal to 1.

2.2.0.2 Matrix Multiplication

To take the matrix product $\mathbf{Z} = \mathbf{XY}$ where $\mathbf{X}$ is an $m \times k$ matrix, $\mathbf{Y}$ is $k \times n$ and $\mathbf{Z}$ is $m \times n$, ones has to compute the elements of z_{ij} as

$$z_{ij} = \sum_{h=1}^{k} x_{ih} y_{hj}.$$

Observe that this is defined only when the number of columns in $\mathbf{X}$ equals the number of rows in $\mathbf{Y}$. Also matrix multiplication is associative so that

$$\mathbf{X}(\mathbf{YZ}) = (\mathbf{XY})\mathbf{Z} = \mathbf{XYZ}.$$

However it is not commutative so that in general $\mathbf{XY} \neq \mathbf{YX}$, even when they are both meaningful.

2.2.0.3 Transpose

The transpose of a matrix $\mathbf{X}$ $m \times n$ is written as $\mathbf{X}^T$ $n \times m$ and is formed by reflecting it around the principal diagonal, so that the k-th column becomes the k-th row and vice-versa. If one wants to compute the transpose of a matrix product $\mathbf{XY}$, note that

$$(\mathbf{XY})^T = \mathbf{Y}^T\mathbf{X}^T.$$

2.2.1 Inverse

A square matrix $\mathbf{X}$ may have an inverse $\mathbf{X}^{-1}$ or not. If it has an inverse, then $\mathbf{I} = \mathbf{X}^{-1}\mathbf{X} = \mathbf{X}\mathbf{X}^{-1}$. If a matrix does not have an inverse, it is called singular. The inverse of the identity matrix is the identity matrix itself. Taking the inverse of a matrix product $\mathbf{XY}$ then, it is possible to equivalently take the inverse of each matrix individually, and reverse the order of multiplication

$$(\mathbf{XY})^{-1} = \mathbf{Y}^{-1}\mathbf{X}^{-1}.$$

2.2.2 Determinant

Each square matrix $\mathbf{X}$ has a scalar determinant denoted by $\det(\mathbf{X})$. A matrix is singular if and only if determinant is zero. For a diagonal matrix the determinant is the product of the diagonal values. It follows that the determinant of $\mathbf{I}$ is 1. Determinants have the following properties:

- $\det(\mathbf{X}^T) = \det(\mathbf{X})$
- $\det(\mathbf{XY}) = \det(\mathbf{X}) \det(\mathbf{Y})$
- $\det(\mathbf{X}^{-1}) = \frac{1}{\det(\mathbf{X})}$ if $\mathbf{X}$ is non singular.

2.2.3 Trace

The trace of a matrix $\mathbf{X}$ is the sum of the diagonal values (the matrix itself needs not to be diagonal). The traces have the following properties:

- $\text{tr}(\mathbf{X}^T) = \text{tr}(\mathbf{X})$
- $\text{tr}(\mathbf{XY}) = \text{tr}(\mathbf{YX})$
- $\text{tr}(\mathbf{X} + \mathbf{Y}) = \text{tr}(\mathbf{Y}) + \text{tr}(\mathbf{X})$
- $\text{tr}(\mathbf{XYZ}) = \text{tr}(\mathbf{ZXY}) = \text{tr}(\mathbf{YZX})$,

where in the last relation the trace is invariant for cyclic permutations only, so that in general $\text{tr}(\mathbf{XYZ}) \neq \text{tr}(\mathbf{XZY})$.

2.2.4 Symmetric positive-definite (SPD) matrices

A $d \times d$ real symmetric matrix $\mathbf{X}$ is positive definite if $y^T\mathbf{X}y > 0$ for all non-zero vectors y. Every positive definite matrix is invertible and its inverse is also positive definite. The determinant and trace of a symmetric positive definite matrix are always positive (they equal the product and the sum of its eigenvalues, respectively).

2.2.5 Singular value decomposition

The singular value decomposition (SVD) is a factorization of a (general) matrix $\mathbf{X}$ $m \times n$ such that $\mathbf{X} = \mathbf{ULV}^T$, where $\mathbf{U}$ $m \times m$ is an orthogonal matrix, $\mathbf{L}$ is a $\mathbf{L}$ $m \times n$ diagonal matrix and $\mathbf{V}$ $n \times n$ is an orthogonal matrix. Note that, if $\mathbf{X}$ is SPD singular value decomposition it is also the eigenvalue decomposition (EVD).

The number of non-zero singular values is called the *rank* of the matrix. The ratio of the smallest to the largest singular value is known as the condition number: it is roughly a measure of how invertible the matrix is.

2.2.6 Symmetric Matrices Vectorization

Given a $d \times d$ symmetric matrix $\mathbf{X}$, it has only $d(d+1)/2$ independent coefficients, which are the upper triangular or lower triangular part of the matrix. Thus, the vector operator is defined as:

$$y = \text{vec}(\mathbf{X}) = [x_{1,1}\ x_{1,2}\ \ldots x_{1,d}\ x_{2,2}\ x_{2,3}\ \ldots x_{d,d}], \tag{2.1}$$

where y is the map of $\mathbf{X} \in \mathbb{R}^m$, with $m = d(d+1)/2$.

2.2.7 Matrix Differentiation

In this thesis I am often called upon to take derivative of matrices. The derivative of a differentiable function $f(\mathbf{X})$ returns to a scalar, with respect to $\mathbf{X}$ is a matrix $\mathbf{Y}$ of the same dimension with elements

$$y_{ij} = \frac{\partial f}{\partial x_{ij}}.$$

Useful cases of matrix differentiation are the following:

Derivative of determinant: $\frac{\partial \det(\mathbf{X})}{\partial \mathbf{X}} = \det(\mathbf{X})\mathbf{X}^{-T}$.
Derivative of log determinant: $\frac{\partial \log(\det(\mathbf{X}))}{\partial \mathbf{X}} = \mathbf{X}^{-T}$.
Derivative of inverse: $\frac{\partial \mathbf{X}^{-1}}{\partial x} = \mathbf{X}^{-1}\frac{\partial \mathbf{X}}{\partial x}\mathbf{X}^{-1}$.
Derivative of trace: $\frac{\partial \operatorname{tr}(\mathbf{X})}{\partial \mathbf{X}} = \left(\frac{\partial(\mathbf{X})}{\partial \mathbf{X}}\right)^{T}$.

2.3 Elements of Topology

Topology is the mathematical area that studies the properties that are preserved through deformations, twistings, and stretchings of shapes and in general of manifolds. To present some concepts of topology I exploited [AMS08, Spe, Ber03].

2.3.1 Topological Space

In this Section the basic definition of a topological space is given.

Definition 1 (Topology) *Let $\mathcal{X}$ an nonempty set. A* topology *on $\mathcal{X}$ is a collection of $\mathcal{T}$ subsets of $\mathcal{X}$ which are termed* opensets *with the following properties:*

- $\mathcal{X}, \emptyset \in \mathcal{T}$ *($\mathcal{X}$ and $\emptyset$ are open).*
- *The union of any collection of sets in $\mathcal{T}$ is in $\mathcal{T}$.*
- *The intersection of any finite number of sets in $\mathcal{T}$ is in $\mathcal{T}$.*

Definition 2 (Topological Space) *A topological space is a pair $(\mathcal{X}, \mathcal{T})$ where $\mathcal{X}$ is a set and $\mathcal{T}$ is a topology on $\mathcal{X}$. When the topology is made clear by the context or is irrelevant, the topological space is simply referred to as $\mathcal{X}$.*

Definition 3 (Continuous Function between Topological Spaces) *Let $(\mathcal{X}, \mathcal{T}_{\mathcal{X}})$ and $(\mathcal{Y}, \mathcal{T}_{\mathcal{Y}})$ two topological spaces. A function $f : \mathcal{X} \to \mathcal{Y}$ is termed* continuous *if*

$$\forall \mathcal{V} \in \mathcal{T}_{\mathcal{Y}}, \mathcal{U} := f^{-1}(\mathcal{V}) \in \mathcal{T}_{\mathcal{X}},$$

where namely, the pre image of any open set in $\mathcal{Y}$ is an open set in $\mathcal{X}$.

Definition 4 (Base of a Topological Space) *Given a topological space $(\mathcal{X}, \mathcal{T})$, a subset $\mathcal{B} \subset \mathcal{T}$ is termed* base *of $\mathcal{T}$ if it is a collection of open sets in $\mathcal{T}$ such that every open set in $\mathcal{T}$ can be written as a union of elements of $\mathcal{B}$*

$$\forall\, \mathcal{A} \in \mathcal{T},\ \mathcal{A} = \bigcup_{\lambda \in \Lambda} \mathcal{B}_{\lambda} (\in \mathcal{B}).$$

If Λ is countable then $\mathcal{B}$ is a countable base.

Definition 5 (Relative Topology) *Let $\mathcal{A} \subset (\mathcal{X}, \mathcal{T})$, the* relative topology *$\mathcal{T}_{\mathcal{A}}$ is naturally defined on $\mathcal{A}$. Given an open set $\mathcal{U} \in \mathcal{T}_{\mathcal{A}}$, if $\mathcal{U} = \mathcal{A} \cap \mathcal{V}$ and $\mathcal{V} \in \mathcal{T}$, then $(\mathcal{A}, \mathcal{T}_{\mathcal{A}})$ is termed* topological subspace *of $(\mathcal{X}, \mathcal{T})$.*

Definition 6 (Product Topology) *Let $(\mathcal{X}, \mathcal{T}_{\mathcal{X}})$ $(\mathcal{Y}, \mathcal{T}_{\mathcal{Y}})$ two topological spaces, $\mathcal{X} \times \mathcal{Y}$ is equipped with the* product topology *$\mathcal{T}_{\mathcal{X}} \times \mathcal{T}_{\mathcal{Y}}$ such that $\mathcal{A} \subset \mathcal{X} \times \mathcal{Y}$ is open if it is the union of* open rectangles *$\mathcal{A}_{\mathcal{X}} \times \mathcal{A}_{\mathcal{Y}}$, where $\mathcal{A}_{\mathcal{X}} \in \mathcal{T}_{\mathcal{X}}$ and $\mathcal{A}_{\mathcal{Y}} \in \mathcal{T}_{\mathcal{Y}}$.*

Definition 7 (homeomorphism) *Let* $f : \mathcal{X} \to \mathcal{Y}$ *is an* homeomorphism *it* f *is* bijective *and* bicontinuous*, i.e. continuous together with its inverse.*

Definition 8 (Homeomorphic Spaces) $(\mathcal{X}, \mathcal{T}_\mathcal{X})$ *and* $(\mathcal{Y}, \mathcal{T}_\mathcal{Y})$ *are* homeomorphic *($\mathcal{X} \approx \mathcal{X}$) if it exists an homeomorphism* $f : \mathcal{X} \to \mathcal{Y}$.

Example 2.1 (The standard topology of the real line). A set $\mathcal{A}$ is called open if it is the union of open intervals (see Fig. 2.1), that can be possibly empty. This is

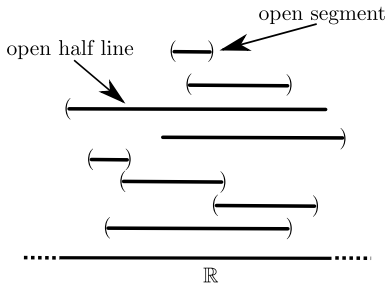


Fig. 2.1. It is easy to see that every open interval is the union of open intervals limited.

equivalent to the following property:

$$\forall \mathbf{X} \in \mathcal{A},\ \exists \mathcal{I} \ni \mathbf{X} \text{ s.t. } \mathcal{I} \subset \mathcal{A}.$$

A graphical representation of the property is depicted in Fig. 2.2. In this example it is easy to see that all the previous conditions are satisfied.

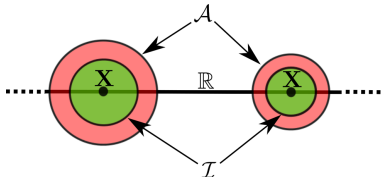


Fig. 2.2. An example of the union of open intervals.

A fundamental concept of geometry is the distance among points. Choosing a distance is equivalent to defining a metric on a space and it can be used to quantify the concepts of near and far. For example, using the Euclidean metric on real line space you can pick a point and say that all points located less than one meter from it are near while the others are far. Thus, the set of all the near points falls into a *ball* of a one-meter radius. This idea can be generalized for arbitrary spaces. The notion of ball allows defines an *open ball*. Colloquially, a set $\mathcal{X}$ is open if any point $\mathbf{X} \in \mathcal{X}$ can be moved in a small neighbourhood $\mathcal{U}_\mathbf{X}$ remaining in $\mathcal{X}$. The notion of an open set provides a fundamental way to speak of nearness of points in a topological space, without having explicitly a concept of distance defined.

2.3.2 Fundamental Properties of a Topological Space

Let $\mathcal{X}$ be a topological space. A subset $\mathcal{A}$ of $\mathcal{X}$ is defined to be *closed* if the set

$$\mathcal{X} - \mathcal{A} := \mathbf{X} \in \mathcal{X} : \mathbf{X} \notin \mathcal{A}$$

is open. A *neighbourhood* of a point $\mathbf{X} \in \mathcal{X}$ is a subset of $\mathcal{X}$ that includes an open set containing $\mathbf{X}$.

Let $\mathcal{X}$ be a topological space. A collection $\mathcal{A} = \{\mathcal{A}_\alpha\}$ ($\alpha \in \mathcal{R}$) with $\mathcal{R}$ any index set, defined as a *covering* of $\mathcal{X}$, if $\bigcup_{\alpha \in \mathcal{R}} \mathcal{A}_\alpha = \mathcal{X}$. The space $\mathcal{X}$ is termed as *compact* if *every* open covering[1] $\mathcal{A}$ of $\mathcal{X}$ contains a finite sub-collection that also covers $\mathcal{X}$. The Heine-Borel theorem says that a subset of $\mathbb{R}^m$ is compact if and only if it is closed and bounded.

A topological space is *second-countable* if it has a countable base.

2.3.3 The Hausdorff separation axiom

In order to start working with manifolds it is necessary to introduce the Hausdorff separation axioms. A topological space is a Hausdorff or T_2 space if $\exists \mathcal{U}_{\mathbf{X}_1} \ni \mathbf{X}_1$, $\mathcal{U}_{\mathbb{X}_1} \not\ni \mathbf{X}_2$, $\exists \mathcal{U}_{\mathbf{X}_2} \ni \mathbf{X}_2$, $\mathcal{U}_{\mathbf{X}_2} \not\ni \mathbf{X}_1$, and $\mathcal{U}_{\mathbf{X}_1} \cup \mathcal{U}_{\mathbf{X}_2} = \emptyset$.

2.4 Manifolds

Definition 9 (Topological Manifold) *A topological space $\mathcal{M}$ is said to be a n-dimensional* topological manifold *if*

- *$\mathcal{M}$ is T_2.*
- *$\mathcal{M}$ admits a countable base.*
- *$\mathcal{M}$ is locally Euclidean such that $\forall \mathbf{X} \in \mathcal{M}$, $\exists \mathcal{U}_{\mathbf{X}} \ni \mathbf{X}$ (a neighbourhood of $\mathbf{X}$), $\exists \mathcal{V} \subset \mathbb{R}^n$, with n fixed, and a homeomorphism $\varphi : \mathcal{U} \to \mathcal{V}$ called* local chart*.*

The composition of one chart with the inverse of another chart is a diffeomorphism of class C^k called a *transition map* (see Fig. 2.3):

$$\varphi_\beta \circ \varphi_\alpha^{-1} : \varphi_\alpha(\mathcal{U}_\alpha \cup \mathcal{U}_\beta) \to \varphi_\beta(\mathcal{U}_\alpha \cup \mathcal{U}_\beta),$$

given $\mathcal{U}_\alpha \cup \mathcal{U}_\beta \neq \emptyset$. A *smooth manifold* is a topological manifold where transition maps are all smooth.

In principle, there are two classes of manifolds that I wish to consider: embedded submanifolds of $\mathbb{R}^m$ and quotient manifolds of $\mathbb{R}^m$. Embedded submanifolds are the easiest to understand, as they have the natural form of an explicit constraints set in $\mathbb{R}^m$. Quotient spaces are more difficult to visualize, as they are not defined as sets of matrices; rather, each point of the quotient space is an equivalence class of $\mathbb{R}^m$. A concrete example is the space of the SPD (or covariance) matrices.

Recalling Definition 9, an m-dimensional manifold can be defined as a set $\mathcal{M}$ covered with a collection of coordinate patches, or charts, that identify certain subsets of $\mathcal{M}$ with open subsets of $\mathbb{R}^m$. So the formal definition of a manifold relies on the concepts of charts and atlases.

[1] $\mathcal{A}$ is called an open covering of $\mathcal{X}$ if its elements are open subsets of $\mathcal{X}$

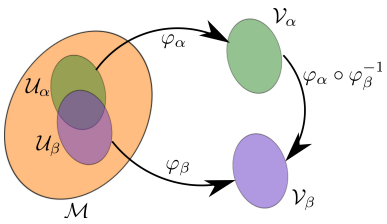


Fig. 2.3. Transition map.

Example 2.2 (The Cartography Case). One of the fundamental problems in is creating a planar map of the Earth depicted in Fig.2.4, which reproduces, in an optimal way, the distances between geographic objects. Mathematically this problem corresponds to the problem of mapping isometrically a sphere to $\mathbb{R}^2$. It is known that, as a consequence of the "Theorema egregium" by Gauss [Ber03], it is impossible to create a map of the Earth that preserves all (geodesic) distances because the Earth is not flat or, more precisely, its Gaussian curvature is positive.

(a) (b)

Fig. 2.4. The cartography problem of creating a planar map of the Earth: the spherical surface of the Earth in (a) is to be mapped into a local chart (b) so that it preserves the geodesic distances.

Let $\mathcal{M}$ be a set. A bijection (one-to-one correspondence) ϕ of a subset $\mathcal{U}$ of $\mathcal{M}$ onto an open subset of $\mathbb{R}^m$ is called a m-dimensional chart of the set $\mathcal{M}$, denoted by $(\mathcal{U}, \phi)$. Given a chart $(\mathcal{U}, \phi)$ and $\mathbf{X} \in \mathcal{U}$, the elements $\phi(\mathbf{X}) \in \mathbb{R}^m$ are called the coordinates of $\mathbf{X}$ in the chart. Thus, it is possible to study objects associated with $\mathcal{U}$ by bringing them to the subset $\phi(\mathcal{U})$ of $\mathbb{R}^m$. For example, if f is a real-valued function on $\mathcal{U}$, then the function composition $f \circ \phi^{-1}$ is a function from $\mathbb{R}^m$ to $\mathbb{R}$, with domain $\phi(\mathcal{U})$, to which methods of real analysis can be applied. To take advantage of this idea, you must require that each point of the set $\mathcal{M}$ be at least in one chart domain. I give an intuitive example considering the space of the SPD tensor commonly known as covariance matrices.

Bringing together all the charts of $\mathcal{M}$ the concept of *atlas* naturally pops out. A C^∞ (smooth, i.e., differentiable for all degrees of differentiation) atlas of $\mathcal{M} \to \mathbb{R}^d$ is a collection of charts $(\mathcal{U}_\alpha, \phi_\alpha)$ of the set $\mathcal{M}$ such that

- $\bigcup_\alpha \mathcal{U}_\alpha = \mathcal{M}$,
- for any pair α, β with $\mathcal{U}_\alpha \cup \mathcal{U}_\beta \neq \emptyset$, the set $\phi_\alpha(\mathcal{U}_\alpha \cup \mathcal{U}_\beta)$ and the set $\phi_\beta(\mathcal{U}_\alpha \cup \mathcal{U}_\beta)$ are open sets in $\mathbb{R}^m$ and the change of coordinates $\phi_\beta \circ \phi_\alpha^{-1} : \mathbb{R}^m \mapsto \mathbb{R}^m$ is smooth.

Given an atlas $\mathcal{A}$, let $\mathcal{A}^+$ be the set of all charts $(\mathcal{U}, \phi)$ such that $\mathcal{A} \cup \{(\mathcal{U}, \phi)\}$ is also an atlas. It is easy to see that $\mathcal{A}^+$ is also an atlas, called the maximal atlas (or complete atlas) generated by the atlas $\mathcal{A}$. Two atlases are equivalent if and only if they generate the same maximal atlas. A maximal atlas of a set $\mathcal{M}$ is also called a *differentiable structure* on $\mathcal{M}$.

Given a manifold $(\mathcal{M}, \mathcal{A}^+)$, a chart of the set $\mathcal{M}$ that belongs to $\mathcal{A}^+$ is called a chart of the manifold $(\mathcal{M}, \mathcal{A}^+)$, and its domain is a coordinate domain of the manifold. With a chart around a point $\mathbf{X} \in \mathcal{M}$, I mean a chart of $(\mathcal{M}, \mathcal{A}^+)$, whose domain $\mathcal{U}$ contains $\mathbf{X}$. The set $\mathcal{U}$ is then a coordinate neighbourhood of $\mathbf{X}$.

A manifold is *connected* if it cannot be expressed as the disjoint union of two nonempty open sets. Equivalently (for a manifold), any two points can be joined by a piecewise smooth curve segment. The connected components of a manifold are open, thus they admit a natural differentiable structure as open submanifolds.

2.4.1 Embedded manifolds

This kind of manifolds is almost everywhere. In fact, all tangible objects can be represented as smooth two-dimensional manifolds residing in the three-dimensional Euclidean space. Such manifolds are often referred as embedded manifolds, as they constitute subspaces of a larger ambient space (the 3-dimensional world) and are embedded into it.

They can be described by a smooth map $\phi : \mathcal{U} \mapsto \mathbb{R}^m$ from a subset $\mathcal{U}$ of $\mathbb{R}^{m-1}$. In this case the set $\mathcal{U}$ is called a *parametrization domain*, while $\mathcal{M} = \phi(\mathcal{U})$ in $\mathbb{R}^m$ is referred to as a *parametric manifold*.

Let $(\mathcal{M}, \mathcal{A}^+)$ and $(\mathcal{N}, \mathcal{B}^+)$ be manifolds such that $\mathcal{N} \subset \mathcal{M}$. The manifold $\mathcal{N}$ is called an immersed submanifold of $\mathcal{M}$ if the inclusion map $f : \mathcal{N} \to \mathcal{M} : \mathbf{X} \mapsto \mathbf{X}$ is an immersion. The concepts of immersion (and submersion) will make it possible to define submanifolds in a concise way. Let $f : \mathcal{M} \to \mathcal{N}$ be a differentiable function from a manifold $\mathcal{M}$ of dimension m_1 into a manifold $\mathcal{N}$ of dimension m_2. Given a point $\mathbf{X}$ of $\mathcal{M}$, the rank of f at $\mathbf{X}$ is the dimension of the range of $d\hat{f}(\phi_1(\mathbf{X})) : \mathbb{R}^{m_1} \to \mathbb{R}^{m_2}$, where $\hat{f} = \phi_2 \circ f \circ \phi_1^{-1}$ (where ϕ_1 and ϕ_2 are charts in a neighbourhood of $\mathbf{X}$) is a coordinate representation of f around $\mathbf{X}$, and $d\hat{f}(\phi_1(\mathbf{X}))$ denotes the differential of $\hat{f}$ at $\phi_1(\mathbf{X})$.The function f is called an *immersion* if its rank is equal to m_1 at each point of its domain (hence $m_1 \leq m_2$). f is a *submersion* and the rank of its differential is m_2, that is, *df* is surjective (therefore $m_2 \geq m_1$).

Since $\mathcal{M}$ and $\mathcal{N}$ are manifolds, they are also topological spaces with their manifold topology. If the manifold topology of $\mathcal{N}$ coincides with its subspace topology induced from the topological space $\mathcal{M}$, then $\mathcal{N}$ is called an *embedded submanifold* of $\mathcal{M}$. In particular, an $\mathcal{N}$ in $\mathbb{R}^N$ is locally a level set of a smooth submersive

function $f : \mathbb{R}^N \to \mathbb{R}^m$ (N is $m + k$ dimensional and k is the dimension of $\mathcal{N}$): without loss of generality, any point $\mathbf{X} \in \mathcal{N}$ has an open neighbourhood given by $f^{-1}(0)$ ($0 \in \mathbb{R}^m$) (obviously $f(\mathbf{X}) = 0$, and $\mathcal{N}$ is endowed with the relative topology inherited from $\mathbb{R}^N$.

2.4.2 Quotient manifolds

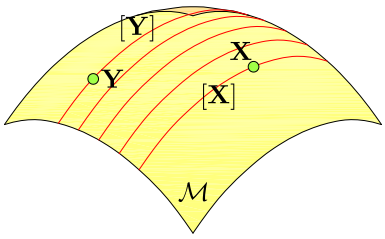


Fig. 2.5. Quotient Manifold.

Intuitively, quotient manifolds are built from suitable identification of points of a given manifold. More formally let $\mathcal{M}$ be a manifold equipped with an equivalence relation (reflexive, symmetric, and transitive) $\sim$. The set

$$[\mathbf{X}] := \{\mathbf{Y} \in \mathcal{M} : \mathbf{X} \sim \mathbf{Y}\}$$

that is of all elements that are equivalent to a point $\mathbf{X}$ is called the equivalence class containing $\mathbf{X}$. The set $\mathcal{M}/\sim := \{[\mathbf{X}] : \mathbf{X} \in \mathcal{M}\}$ of all equivalence classes of $\sim$ in $\mathcal{M}$ is called the quotient of $\mathcal{M}$ by $\sim$. The mapping $\pi : \mathcal{M} \to \mathcal{M}/\sim$ defined by $\mathbf{X} \mapsto [\mathbf{X}]$ is called the *natural projection* or *canonical projection*.

Let $(\mathcal{M}, \mathcal{A}^+)$ be a manifold with an equivalence relation $\sim$ and let $\mathcal{B}^+$ be a manifold structure on the set $\mathcal{M}/\sim$. The manifold $(\mathcal{M}/\sim, \mathcal{B}^+)$ is called a *quotient manifold* of $(\mathcal{M}, \mathcal{A}^+)$ if the natural projection π is a submersion (see the previous Subsection for details). Examples of quotient manifolds are the real projective space, Grassmann manifolds and the space of SPD matrices. To "visualize" the concept of quotient manifold, an example is depicted in Fig. 2.5.

When $\mathcal{M}$ is $\mathbb{R}^{d\times d}$ or a sub-manifold (or embedded manifold) of $\mathbb{R}^{d\times d}$, I call $\mathcal{M}/\sim$ a *matrix quotient manifold*. I call a *matrix manifold* any manifold that is constructed from $\mathbb{R}^{d\times d}$ by the operations of taking embedded sub-manifolds and quotient manifolds.

2.4.3 Tangent vectors and tangent spaces

At each point of a differentiable manifold $\mathcal{M}$ you can draw a *tangent space* (tangent plane in the 2D case). Tangent spaces (see Fig. 2.6) are of fundamental importance in the theory of differentiable manifolds, because they are needed if one wishes to make sense of the notion of the directional derivative. Intuitively speaking, you think of the tangent space at a point as the best linear approximation of the manifold near that point.

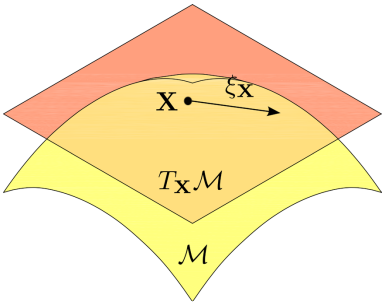


Fig. 2.6. Tangent vectors and tangent spaces.

Let $\mathcal{M}$ be a manifold. A smooth mapping $\gamma : \mathbb{R} \to \mathcal{M} : t \mapsto \gamma(t)$ is called a curve in $\mathcal{M}$. The idea of defining a derivative

$$\dot{\gamma}(t) := \lim_{r \to 0} \frac{\gamma(t+r) - \gamma(t)}{r}$$

requires a vector space structure to compute the difference $\gamma(t+r) - \gamma(t)$ thus fails for an abstract nonlinear manifold. To overcome this problem one proceed as follows, given a smooth realvalued function f on $\mathcal{M}$, the function

$$f \circ \gamma : t \mapsto f(\gamma(t))$$

is a smooth function from $\mathbb{R}$ to $\mathbb{R}$ with a well-defined classical derivative.

Let $\mathcal{F}_{\mathbf{X}(\mathcal{M})}$ denote the set of smooth real-valued functions defined on a neighbourhood of $\mathbf{X}$. A tangent vector $\xi_{\mathbf{X}}$ to a manifold $\mathcal{M}$ at a point $\mathbf{X}$ is a mapping from $\mathcal{F}_{\mathbf{X}(\mathcal{M})}$ to $\mathbb{R}$ such that there exists a curve γ on $\mathcal{M}$ with $\gamma(0) = \mathbf{X}$ which satisfies

$$\xi_{\mathbf{X}} = \dot{\gamma}(0)f := \frac{d(f(\gamma(t)))}{dt},$$

where $t = 0$ for all $f \in \mathcal{F}_{\mathbf{X}(\mathcal{M})}$. Such a curve γ is said to realize the tangent vector $\xi_{\mathbf{X}}$. The *tangent space* to $\mathcal{M}$ at $\mathbf{X}$, denoted by $T_{\mathbf{X}}\mathcal{M}$, is the set of all the tangent vectors to $\mathcal{M}$ at $\mathbf{X}$. This set admits a structure of vector space as follows. Given $\dot{\gamma}_1(0)$ and $\dot{\gamma}_2(0)$ in $T_{\mathbf{X}}\mathcal{M}$ and $a, b \in \mathbb{R}$, define

$$(a\dot{\gamma}_1(0) + b\dot{\gamma}_2(0))f := a(\dot{\gamma}_1(0)f) + b(\dot{\gamma}_2(0)f).$$

In other words, the tangent vectors (the elements of $T_{\mathbf{X}}\mathcal{M}$) are the "velocities" of the curves in $\mathcal{M}$ issuing from $\mathbf{X} \in \mathcal{M}$ or, equivalently, the "directional derivatives" of the smooth functions defined in a neighbourhood of $\mathbf{X}$.

2.5 Riemannian Geometry

As we have seen, tangent vectors on manifolds generalize the notion of a directional derivative, but to define the concept of length of a curve it is necessary to make a step further. This is done by endowing every tangent space $T_{\mathbf{X}}\mathcal{M}$ with an

inner product $\langle . \rangle_{\mathbf{X}}$, i.e., a bilinear, symmetric positive-definite form and the inner product induces a norm

$$||\xi_{\mathbf{X}}||_{\mathbf{X}} := \sqrt{\langle \xi_{\mathbf{X}}, \xi_{\mathbf{X}} \rangle_{\mathbf{X}}},$$

and therefore a distance between points on $\mathcal{M}$.

A manifold whose tangent spaces are endowed with a smoothly varying inner product is called a *Riemannian manifold*. The smoothly varying inner product is called the *Riemannian metric* g. I will use interchangeably the notation

$$g(\xi_{\mathbf{X}}, \zeta_{\mathbf{X}}) = g_{\mathbf{X}}(\xi_{\mathbf{X}}, \zeta_{\mathbf{X}}) = \langle \xi_{\mathbf{X}}, \zeta_{\mathbf{X}} \rangle = \langle \xi_{\mathbf{X}}, \zeta_{\mathbf{X}} \rangle_{\mathbf{X}}$$

to denote the inner product of two elements $\xi_{\mathbf{X}}$ and $\zeta_{\mathbf{X}}$ of $T_{\mathbf{X}}\mathcal{M}$. A vector space endowed with an inner product is a particular Riemannian manifold called *Euclidean space*. It is worth noting that any manifold admits a Riemannian structure.

The Riemannian distance on a connected Riemannian manifold $\mathcal{M}$ is

$$d : \mathcal{M} \times \mathcal{M} \to \mathbb{R} : d(\mathbf{X}, \mathbf{Y}) = \inf_{\Gamma} L(\gamma),$$

where Γ is the set of all curves in $\mathcal{M}$ joining the points $\mathbf{X}$ and $\mathbf{Y}$, $\gamma \in \Gamma$, and $L(\gamma) = \int_a^b \sqrt{\langle \dot{\gamma}(t), \dot{\gamma}(t) \rangle}$. It can be shown that the Riemannian distance defines a *metric*; i.e.,

(positive-definiteness) $d(\mathbf{X}, \mathbf{Y}) \geq 0$, with $d(\mathbf{X}, \mathbf{Y}) = 0$ iff $\mathbf{X} = \mathbf{Y}$;
(symmetry) $d(\mathbf{X}, \mathbf{Y}) = d(\mathbf{Y}, \mathbf{X})$;
(triangle inequality) $d(\mathbf{X}, \mathbf{Z}) + d(\mathbf{Z}, \mathbf{Y}) \geq d(\mathbf{X}, \mathbf{Y})$.

2.5.1 Exponential mapping

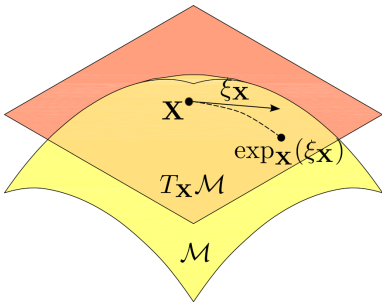


Fig. 2.7. Exponential Mapping.

In Riemannian geometry the exponential mapping (see Fig. 2.7) can be thought as a way to a map point $\xi_{\mathbf{X}}$ from one of the tangent spaces $T_{\mathbf{X}}\mathcal{M}$ to $\mathcal{M}$. To introduce the exponential mapping, it is necessary to define a geodesic curve. A *geodesic* γ on a Riemannian manifold $\mathcal{M}$ is a smooth parametric curve, with parameter proportional to the arc-length s induced by the metric ($ds = \sqrt{g(\xi_{\mathbf{X}}, \zeta_{\mathbf{X}})}\, dt$) which (locally) minimises the distance between two points on $\mathcal{M}$ (the latter given by the infimum of the lengths of all curves connecting the points in question. They are

also given by the parallel curves (with respect to the Levi-Civita connection), but I shall not elaborate on this point any further, since this is not strictly needed for our purposes.

For every $\xi \in T_{\mathbf{X}}\mathcal{M}$, exists a unique geodesic $\gamma(t; \mathbf{X}, \xi) : I \to \mathcal{M}$ such that $\gamma(0) = \mathbf{X}$ and $\dot{\gamma}(0) = \xi$. Moreover, you have the homogeneity property $\gamma(t; \mathbf{X}, \xi) := \gamma(at; \mathbf{X}, \xi)$. The mapping

$$\exp_{\mathbf{X}} : T_{\mathbf{X}}\mathcal{M} \to \mathbf{X} : \xi \mapsto \exp_{\mathbf{X}} \xi = \gamma(1; \mathbf{X}, \xi)$$

is called the *exponential map* at $\mathbf{X}$. When the domain of definition of $\exp_{\mathbf{X}} \mathcal{M}$ is the whole $T_{\mathbf{X}}\mathcal{M}$ for all $\mathbf{X} \in \mathcal{M}$, the manifold $\mathcal{M}$ is termed (geodesically) *complete*. It can be shown that $\exp_{\mathbf{X}}$ defines a local diffeomorphism (smooth bijection) of a neighbourhood of $0_{\mathbf{X}} \in T_{\mathbf{X}}\mathcal{M}$ onto a neighbourhood $\mathcal{U}_{\mathbf{X}}\mathcal{M}$, therefore, in general, it is not a global diffeomorphism.

2.5.2 Lie Groups

Lie groups are, by definition, are both groups and manifolds and their group operations are differentiable. Group operations are then automatically C^∞. Examples of Lie groups are $Gl(n, \mathbb{R})$ (the general linear group), $O(n)$ (the orthogonal group), $SO(n)$ (special orthogonal group), $O(p, q)$, and $SO(p, q)$.

2.5.3 Homogeneous Spaces

Many geometric objects appear, not as a Lie group, but as a quotient of a Lie group by a subgroup. Such objects are called *homogeneous spaces*. Consider, for example, the set of all k-dimensional vector subspaces of $\mathbb{R}^m$. I denote this set by $Grass(k, d)$ and call it a Grassmann manifold. It is also a quotient of Lie groups. Introduce the orthogonal group $O(n)$ of all linear isometries of $\mathbb{R}^m$, so that $Grass(k, d) = O(d)/(O(k) \times O(d - k))$. Another important group which is also a quotient manifold is Sym_d^+, composed by the SPD matrices. It can be defined as $Sym_d^+ = Gl(n, \mathbb{R})/O(d, \mathbb{R})$. The name homogeneous space expresses their fundamental property: colloquially, each point sees the same landscape.

Recall that quotient manifolds are defined by equivalence relations. In the case G/H of a Lie group G divided by some subgroup H, the manifold character of the quotient is guaranteed as long as the subgroup H is closed in G. Moreover, compact subgroup(s) H will produce Riemannian geometry on the quotient. Assuming a Riemannian metric g on G and wants to push it down to the quotient G/H. Thus, you need that g is invariant under the action of H that is called the *isotropy group* of G/H.

Let G any Lie group and consider the *left translation map* $\lambda_{\mathbf{Y}} : G \to G : \mathbf{X} \mapsto \mathbf{YX}$ as in Fig. 2.8. It is a diffeomorphism by definition[2], so its differential $d\lambda_{\mathbf{Y}}$ is a linear isomorphism between the two tangent spaces $T_{\mathbf{X}}G$ and $T_{\mathbf{YX}}G$. Picking any Euclidean structure (any positive definite quadratic form) on $T_{\mathbf{I}}G$ (the tangent space to G at the identity element $\mathbf{I}$) and transporting it to $T_{\mathbf{Y}}G$ by demanding

[2] A (smooth) diffeomorphism $f : \mathcal{M}_1 \to \mathcal{M}_2$ is a bijection such that f and its inverse are both smooth.

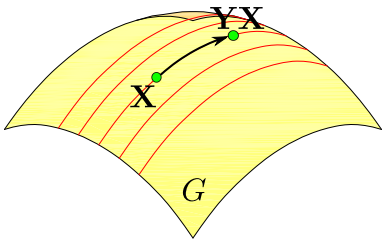


Fig. 2.8. Left Translation Map.

that $d\lambda_{\mathbf{Y}}$ be an isometry, namely a mapping which preserve the distance among points, a Riemannian metric can be built. In particular, doing this for every $\mathbf{Y} \in G$, one gets a Riemannian metric on G which is invariant under left multiplication, and thus it is called *left invariant*. In general it will not be also right invariant. When G is compact, a bi-invariant Riemannian metric always exists.

2.5.4 Symmetric Spaces

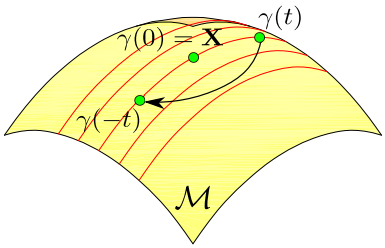


Fig. 2.9. A Symmetric Space.

A symmetric space is a connected Riemannian manifold. *Locally symmetric spaces* are defined as the Riemannian manifolds for which the local geodesic symmetry around any point is a local isometry. This symmetry around $\mathbf{X}$, denoted by $i_{\mathbf{X}}$, is defined as the (geodesic) map changing $\gamma(t)$ into $\gamma(-t)$ for every geodesic γ through $\mathbf{X} = \gamma(0)$, depicted in Fig. 2.9; i.e. geodesic are indefinitely extendible. If the manifold is complete, then one can define a global symmetry $i_{\mathbf{X}} : \mathcal{M} \to \mathcal{M}$ and the manifold $\mathcal{M}$ is then called symmetric if all the $i_{\mathbf{X}}$ are isometries. Thus $\mathcal{M}$ is *symmetric*.

Example 2.3. This example shows the effect of composing isometries on a symmetric space. Let γ a geodesic curve, $i_{\mathbf{X}}$ and $i_{\mathbf{Y}}$ two isometries, and $\gamma(0) = \mathbf{X}$, $\gamma(c) = \mathbf{Y}$. If $\gamma(t)$ and $\gamma(t+2c)$ are defined, then $i_{\mathbf{X}} i_{\mathbf{Y}}(\gamma(t)) = \gamma(t+2c)$ as depicted in Fig. 2.10

Here I point out some interesting facts about symmetric spaces. First, if $\mathcal{M}$ is a symmetric space then $\mathcal{M}$ is complete. Moreover $i_{\mathbf{X}}$ is univocally defined. Second,

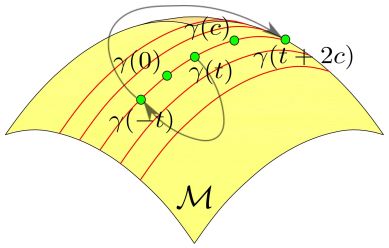


Fig. 2.10. Composition of isometries on a symmetric space.

if $\mathcal{M}$ is simply connected[3] and locally symmetric, then it is (globally) symmetric. Third, imagine to remove a point from $\mathcal{M}$. So, $\mathcal{M}$ is no longer complete and therefore symmetric, but it is still locally symmetric. Last but not least, Lie groups can be symmetric spaces. Let G a Lie group with a (bi)invariant metric, then G is a symmetric space.

To conclude I relate geodesics to one parameter group. It could be defined as a *one-parameter group*. In fact, let $\gamma : \mathbb{R} \to G$ geodesic and $\gamma(0) = \mathbf{I}$. Then

$$i_{\gamma(t)} i_{\mathbf{I}}(\gamma(u)) = \gamma(u + 2t),$$

but $i_{\gamma(t)} i_{\mathbf{I}}(\sigma) = \gamma(t)\sigma\gamma(t)$, then $i_{\mathbf{I}}(\sigma) = \sigma^{-1}$ and

$$i_{\gamma(t)}(\sigma^{-1}) = \gamma(t)\sigma\gamma(t).$$

Therefore

$$\gamma(u + 2t) = \gamma(t)\gamma(u)\gamma(t),$$

and if $u = 0$ $\gamma(2t) = \gamma(t)^2$, so $\gamma(nt) = \gamma(t)^n$. It follows that

$$\gamma(t_1 + t_2) = \gamma(t_1)\gamma(t_2).$$

If t_1/t_2 is rational, the hypothesis always holds. Vice versa, given a one-parameter group, it coincides with the geodesic starting from the identity with the same velocity vector.

2.6 Cases of Interest

This Section contains two case studies of matrix manifolds: Sym^+ and $Grass(p, n)$ which are exploited in this thesis for practical applications.

2.6.1 Sym^+ and Sym

Sym^+, the space of the SPD matrices, can be turned into is a *symmetric Riemannian manifold*. More precisely, Sym^+ can be built as a quotient manifold G/H, where $G = GL(n, \mathbb{R})$ and $H = O(n)$. Given a point $\mathbf{X}$ in Sym^+, a map φ, and its

[3] A connected topological space is simply connected if any loop (with a base point $\mathbf{P}$) can be continuously shrunk to the constant loop $\mathbf{P}$ (abuse of notation)

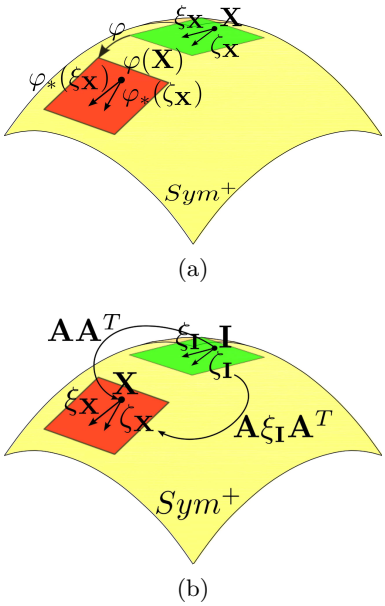


Fig. 2.11. Metric Invariance of Sym^+.

differential φ_* (also termed push-forward map) between two tangent spaces, then the (Euclidean) metric on the tangent spaces is invariant and can be written as follow (see Fig. 2.11(a)):

$$\langle \xi_{\mathbf{X}}, \zeta_{\mathbf{X}} \rangle_{\mathbf{X}} = \langle \varphi_*(\xi_{\mathbf{X}}), \varphi_*(\zeta_{\mathbf{X}}) \rangle_{\varphi(\mathbf{X})}.$$

Given $\mathbf{A} \in O(n)$, the action of H on Sym^+ is shown in the following case:

$$\mathbf{I} \in Sym^+ \mapsto \mathbf{A}\mathbf{I}\mathbf{A}^T = \mathbf{A}\mathbf{A}^T = \mathbf{X} \in Sym^+,$$

which correspond to the following mapping between tangent spaces

$$\zeta_{\mathbf{I}} \mapsto \mathbf{A}\zeta_{\mathbf{I}}\mathbf{A}^T = \zeta_{\mathbf{X}},$$

as shown in Fig. 2.11(b). From the last equation one derives that

$$\zeta_{\mathbf{I}} = A^{-1}\zeta_{\mathbf{X}}A^{-T}.$$

To measure the distance between pair of elements on $T_{\mathbf{I}}Sym^+$ the usual Frobenius (Euclidean) norm

$$\langle \xi_{\mathbf{I}}, \zeta_{\mathbf{I}} \rangle_{\mathbf{I}} := \mathrm{tr}(\xi_{\mathbf{I}}\zeta_{\mathbf{I}}^T)$$

is adopted. By imposing $\langle \xi_{\mathbf{I}}, \zeta_{\mathbf{I}} \rangle_{\mathbf{I}} = \langle \xi_{\mathbf{X}}, \zeta_{\mathbf{X}} \rangle_{\mathbf{X}}$, one obtain that

$$\begin{aligned}
\langle \xi_{\mathbf{X}}, \zeta_{\mathbf{X}} \rangle_{\mathbf{X}} &= \mathrm{tr}(\mathbf{A}^{-1}\xi_{\mathbf{X}}\mathbf{A}^{-T}\mathbf{A}^{-1}\zeta_{\mathbf{X}}\mathbf{A}^{-T}) \\
&= \mathrm{tr}(\mathbf{A}^{-1}\xi_{\mathbf{X}}(\mathbf{A}\mathbf{A}^T)^{-1}\zeta_{\mathbf{X}}\mathbf{A}^{-T}) \\
&= \mathrm{tr}(\mathbf{A}^{-T}\mathbf{A}^{-1}\xi_{\mathbf{X}}(\mathbf{A}\mathbf{A}^T)^{-1}\zeta_{\mathbf{X}}) \\
&= \mathrm{tr}(\mathbf{A}\mathbf{A}^T)^{-1}\xi_{\mathbf{X}}(\mathbf{A}\mathbf{A}^T)^{-1}\zeta_{\mathbf{X}}) \\
&= \mathrm{tr}(\mathbf{X}^{-1}\xi_{\mathbf{X}}\mathbf{X}^{-1}\zeta_{\mathbf{X}}).
\end{aligned}$$

If $\zeta_{\mathbf{X}} = \xi_{\mathbf{X}}$

$$||\xi_{\mathbf{X}}||^2_{\mathbf{X}} = \mathrm{tr}((\mathbf{X}^{-1}\xi_{\mathbf{X}})^2),$$

which is the Riemannian metric on Sym^+.

The geodesic curves on Sym^+ are one-parameter (sub)groups. In particular

$$\begin{aligned}\gamma(t; \mathbf{X}, \xi_{\mathbf{X}}) &= \exp(t\mathbf{A})\mathbf{X}\exp((t\mathbf{A})^T) \\ &= \underbrace{\exp(t\mathbf{A})}_{1+t\mathbf{A}+\ldots}\mathbf{X}\underbrace{\exp(t\mathbf{A}^T)}_{1+t\mathbf{A}^T+\ldots} \\ &= \mathbf{X} + t(\underbrace{\mathbf{A}\mathbf{X} + \mathbf{X}\mathbf{A}^T}_{\xi_{\mathbf{X}}}),\end{aligned}$$

thus

$$\xi_{\mathbf{X}} = \mathbf{A}\mathbf{X} + \mathbf{X}\mathbf{A}^T.$$

If $\mathbf{X} = \mathbf{I}$, then $\xi_{\mathbf{X}} = \mathbf{A} + \mathbf{A}^T$, moreover, if $\xi_{\mathbf{I}}$ is known, one can define $\mathbf{A} = 1/2\xi_{\mathbf{I}}$, so

$$\gamma(t; \mathbf{I}, \xi_{\mathbf{I}}) = \exp(\frac{t}{2}\xi_{\mathbf{I}})\exp(\frac{t}{2}\xi_{\mathbf{I}}) = \exp(\xi_{\mathbf{I}}).$$

Finally, it is worth noting the relation between Sym^+ and Sym, the space of the symmetric matrices. Sym is the vector space of real symmetric matrices. By definition

$$T_{\mathbf{X}}Sym^+ := Sym \quad \forall\, \mathbf{X} \in Sym^+.$$

In fact, the tangent space of Sym^+ at any point, is Sym, the space of symmetric matrices. Indeed, let us consider an interval $J \subset R$ containing 0, and let us consider a smooth curve of matrices $J \ni t \mapsto \gamma(t) \in Sym^+$ with $\Sigma(0) = \mathbf{I}$. Its "velocity" at $\mathbf{I}$, namely $\dot{\gamma}(0)$, belongs to Sym, since the derivative of $\gamma(t)$ is still a symmetric matrix. Vice versa, given a matrix $\xi_{\in}Sym$, it is possible to find a curve in Sym^+ starting at $\mathbf{I}$ with velocity given by $\xi = \dot{\gamma}(0)$. Taking for instance $\gamma(t) = \exp(t\xi)$, if we diagonalize the matrix W and denote its eigenvalues by w_i, $i = 1, 2, \ldots, d$, then the eigenvalues of $\gamma(t)$ are $\exp(tw_i) > 0$, $i = 1, 2, \ldots, d$. Therefore, the matrix is positive definite. By continuity, any curve with the same velocity at $\mathbf{I}$ is locally in Sym^+.

2.6.2 *Grass*(p, n)

Let n be a positive integer and let p be a positive integer not greater than n. Let $Grass(p, n)$ denote the set of all p-dimensional subspaces of $\mathbb{R}^n$, it can be endowed a matrix manifold structure. In particular, The $Grass(p, n)$ is a $p\times(np)$-dimensional compact Riemannian manifold derived as a quotient space of orthogonal groups

$$Grass(p, n) = O(n)/O(p) \times O(n - p),$$

where $O(m)$ is the group of $m \times m$ orthonormal matrices.

Let $\sim$ denote the equivalence relation on $\mathbb{R}^n$ defined by

$$\mathbf{X} \sim \mathbf{Y} \iff \mathrm{span}(\mathbf{X}) = \mathrm{span}(\mathbf{Y}),$$

where span($\mathbf{X}$) denotes the subspace $\{\mathbf{X}\alpha : \alpha \in \mathbb{R}^p\}$ spanned by the columns of $\mathbf{X}$. The set of all matrix representations of span($\mathbf{X}$) is the equivalence relation necessary to define the quotient manifold and can be denoted as

$$\pi^{-1}(\pi(\mathbf{X})) = \{\mathbf{XM} : \mathbf{M} \in \mathbb{R}^{p\times p}\}.$$

If a matrix $\mathbf{X}$ and a subspace $\mathcal{X}$ satisfy $\mathcal{X} = \text{span}(\mathbf{X})$, one can say that $\mathcal{X}$ is the span of $\mathbf{X} \in \mathbb{R}^{n\times p}$, or that $\mathbf{X}$ is the matrix representation of $\mathcal{X}$.

The associated Riemannian metric with $Grass(p, n)$ can be used to scalar product between $\mathbf{X}$ and $\mathbf{Y}$ at $\mathbf{Z}$ as

$$\langle \mathbf{X}, \mathbf{Y}\rangle_{\mathbf{Z}} = \frac{\langle \mathbf{X}, \mathbf{Z}\rangle}{||\mathbf{Z}||^2}.$$

In order to show that $Grass(p, n)$ admits a structure of a Riemannian quotient manifold, it is necessary to show that

$$\langle \xi_{\mathbf{YM}}, \zeta_{\mathbf{YM}}\rangle_{\mathbf{YM}} = \langle \xi_{\mathbf{Y}}, \zeta_{\mathbf{Y}}\rangle_{\mathbf{Y}}$$

for all $\mathbf{M} \in \mathbb{R}^{p\times p}$ as follows. Given $\mathbf{Y} \in \mathbb{R}^{n\times p}$ and $\xi \in T_{\mathcal{Y}}Grass(p, n)$ $(n \times p)$, then

$$\xi_{\mathbf{YM}} = \xi_{\mathbf{Y}}\mathbf{M}, \quad \forall \mathbf{M} \in \mathbb{R}^{p\times p}.$$

Using the last equation

$$\begin{aligned}
\langle \xi_{\mathbf{YM}}, \zeta_{\mathbf{YM}}\rangle_{\mathbf{YM}} &= \langle \xi_{\mathbf{Y}}\mathbf{M}, \zeta_{\mathbf{Y}}\mathbf{M}\rangle_{\mathbf{YM}} \\
&= \text{tr}\left(((\mathbf{YM})^T\mathbf{YM})^{-1}(\xi_{\mathbf{Y}}\mathbf{M})^{-1}(\xi_{\mathbf{Y}}\mathbf{M})^T(\zeta_{\mathbf{Y}}\mathbf{M})\right) \\
&= \text{tr}\left(\mathbf{M}^{-1}(\mathbf{Y}^T\mathbf{Y})^{-1}\mathbf{M}^{-T}\mathbf{M}^T\xi_{\mathbf{Y}}^T\zeta_{\mathbf{Y}}\mathbf{M}\right) \\
&= \text{tr}\left((\mathbf{Y}^T\mathbf{Y})^{-1}\xi_{\mathbf{Y}}^T\zeta_{\mathbf{Y}}\mathbf{M}\right) \\
&= \langle \xi_{\mathbf{Y}}, \zeta_{\mathbf{Y}}\rangle_{\mathbf{Y}}.
\end{aligned}$$

This shows that $Grass(p, n)$, endowed with the Riemannian metric

$$\langle \xi, \zeta\rangle_{\mathcal{Y}} := \langle \xi_{\mathbf{Y}}, \zeta_{\mathbf{Y}}\rangle_{\mathbf{Y}}$$

is a Riemannian quotient manifold of $\mathbb{R}^{n\times p}$.

3

Discriminative Models

Contents

3.1 Introduction

To solve a computer vision problem one needs three components. (1) A model that relates the visual data $\mathbf{x}$ and the world state c mathematically. The model specifies a family of possible relationships between $\mathbf{x}$ and c and it is governed by the model parameters Θ. (2) A learning algorithm that allows to fit Θ using paired training examples $\{\mathbf{x}_i, c_i\}_{i=1,\dots,N}$ where N is the number of training examples. (3) An inference method that takes a new observation $\mathbf{x}$ and uses the model to return the posterior $P(c|\mathbf{x}, \Theta)$ over the world state c.

In this thesis, the model, which is the most important ingredient, is fixed. The *discriminative models* are chosen. These models are used to obtain $P(c|\mathbf{x}, \Theta)$ directly, that means to learn a direct map from inputs $\mathbf{x}$ to the world states (or class label) c. There are several reasons for choosing discriminative rather than generative models (that lean the joint probability $P(c, \mathbf{x})$) listed by Vapnik [Vap98]. To summarise, he says that "one should solve the learning problem directly and

never solve a more general problem as an intermediate step (such as learn $P(\mathbf{x}|c)$ as done by generative models)".

Discriminative models learn the posterior probability $P(c|\mathbf{x}, \Theta)$ making the distribution parameters a probabilistic function of the data $\mathbf{x}$ and the parameters Θ. Therefore the goal of a (supervised) learning algorithm is to fit the parameters Θ using paired training couples $\{\mathbf{x}_i, c_i\}_{i=1,\ldots,N}$ where $c_i \in C$, i.e. the set of all the classes of the problem. This can be done using either the maximum likelihood (ML) approach, or the maximum a posteriori (MAP) one or the Bayesian one [Bel06, HTF11, Pri12].

In this Chapter, the machine learning techniques to train the adopted discriminative models are described.To be more precise, in Sec. 3.2 *boosting* methods are presented and in particular an approach of this family termed *LogitBoost*, inspired by [TPM08]. This because it is one of the most successful works that combine LogitBoost and tensor representation to describe the visual objects. Then, I describe the *Random Forest* framework [Bre01] for its neutral capacity to manage multi-class classification problems and their robustness and efficiency.

Since boosting and random forests (RFs) are strictly related, at this introductory level it is interesting to give a picture of which is the relation between these approaches, as done in Fig. 3.1, inspired by [KSB]. In this Figure, there are three

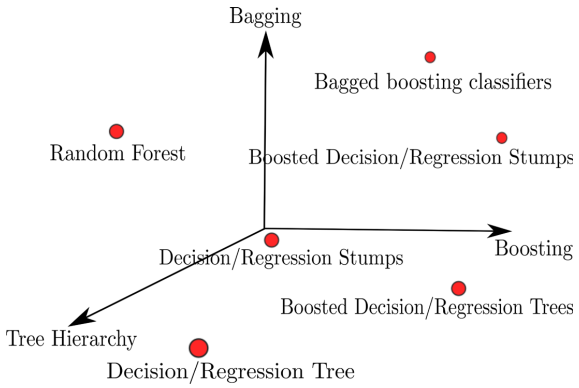


Fig. 3.1. Boosting and Tree-structured Classifiers.

main actors (depicted as the axis): Bagging, Boosting, and Tree Hierarchy. Bagging, or *bootstrap aggregation* [Bre96], is a technique used to reduce the variance of an estimated prediction function. Bagging seems to work particularly well for high-variance, low-bias procedures, such as trees. For regression or "continuous" classification problems, one can simply fit the same regression tree many times, to bootstrap sampled versions of the training data, and average the result. For classification, each committee of trees casts a vote for the predicted class. Boosting was initially proposed as a committee method as well, although, unlike bagging, the committee of weak learners evolves over time, and the members cast a weighted vote. Boosting appears to dominate bagging on most problems, and became the preferred option. RF [Bre01] is a substantial modification of bagging that builds a large collection of decorrelated trees, and then averages them. On many problems

the performance of RF is very similar to boosting, and they are simpler to train and tune. As a consequence, RF is popular.

Sec. 3.4, with relation to the support vector machines (SVMs) classification, shows how to learn naturally from tensor data using *kernel methods*. This family of methods allows to deal with widely different kinds of input examples (i.e. tensors), so it overcomes the limitation of the previous machine learning approaches (Boosting and RF) which are designed to deal with vector examples and can be used with tensors only "reducing" tensors to vectors.

Finally, Sec, 3.5, as done for the previous Chapter, instantiates the presented approaches to some cases of interest.

3.2 Boosting

Boosting is one of the most powerful learning ideas introduced in the last twenty years. It was originally designed for classification problems and can be extended profitably to regression as well. The motivation for boosting was a procedure that combines the outputs of many "weak" classifiers to produce a powerful committee. I begin by describing the most popular boosting algorithm, called *AdaBoost.M1* (or just AdaBoost), to give a rough idea of how boosting works. Consider a two-class problem, with the output variable coded as $C \in \{-1, 1\}$. Given a vector of features $\mathbf{x}$, a classifier $G(\mathbf{x})$ produces a prediction taking one of the two values $\{-1, 1\}$. The error rate on the training sample is

$$\epsilon = \frac{1}{N} \sum_{i=1}^{N} 1\{c_i \neq G(\mathbf{x}_i)\},$$

where $1\{\cdot\}$ is an indicator function. A weak classifier is one whose error rate is only slightly better than random guessing. The purpose of boosting is to apply the weak classification algorithm sequentially updating the weight of each example after every round, thereby producing a sequence of weak classifiers $G_m(X), m = 1, 2, \ldots, M$. The predictions from all of them are then combined through a weighted majority vote to produce the final prediction:

$$G(\mathbf{x}) = \text{sign}\left(\sum_{m=1}^{M} \alpha_m G_m(\mathbf{x})\right).$$

Here $\alpha_1, \alpha_2, \ldots, \alpha_M$ are computed by the boosting algorithm, and weight the contribution of each respective $G_m(\mathbf{x})$. Their effect is to give higher influence to the more accurate classifiers in the sequence.

Alg. 1 shows a schematic of the AdaBoost procedure. Initially, all of the weights are set to $w_i = 1/N$, so that the first step simply trains the classifier on the data in the usual manner. For each successive iteration $m = 2, 3, \ldots, M$, the observation weights are individually modified and the classification algorithm is reapplied to the weighted observations. At step m, the observations that were misclassified by the classifier $G_{m-1}(\mathbf{x})$ induced at the previous step have their weights increased, whereas the weights are decreased for those that were classified

Algorithm 1: AdaBoost

Data: A dataset of N couples $\{(\mathbf{x}_1, c_1), \ldots, (\mathbf{x}_N, c_N)\}$, $c_i \in \{-1, 1\}$.
Result: The classifier G.
begin
  Initialize the observation weights $w_i = 1/N, i = 1, 2, \ldots, N$;
  for $m = 1, 2, \ldots, M$ **do**
    Fit a classifier $G_m(\mathbf{x})$ to the training data using weights w_i;
    Compute
$$\epsilon_m = \frac{\sum_{i=1}^{N} w_i 1\{c_i \neq G(\mathbf{x}_i)\}}{\sum_{i=1}^{N} w_i};$$
    Compute $\alpha_m = \log\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$;
    Set $w_i = w_i \exp[\alpha_m 1\{c_i \neq G_m(\mathbf{x}_i)\}, i = 1, 2 \ldots, N$;

correctly. Therefore, as the iterations proceed, the observations, that are difficult to classify correctly, receive ever-increasing influence. Each successive classifier is thereby forced to concentrate on those training observations that are missed by the previous ones in the sequence. Many variations of AdaBoost are formalised in a unified gradient descent procedure, proposed in [MBB00]. A recent advancement in the theory of boosting is represented by [WSJ+11], where the concept of Equilibrium margin (Emargin) is given. It suggests not only that the minimum margin, that means a high classification confidence, may be inessential for the generalization error [HTF11], but also that a large Emargin and a small empirical error imply a smaller bound of the generalization error.

In the follows, after a state-of-the-are characterization of boosting approaches, the binary and multi-class logistic regression boosting (LogitBoost), which are the probabilistic versions of Boosting, are described.

3.2.1 State of The Art of Boosting Methods

First of all it is necessary to highlight that in the past few years a lot of new boosting frameworks have been presented, thus being impossible to report all of them. See the chosen references to be introduced to boosting and start to deal with the most popular state-of-the-art approaches. [Bel06, Sch02] are deep introductions in boosting theory, while in [MR03, Pri12] a good (but more essential) theoretical introduction is provided, in addition to the state-of-the-art approaches. Moreover, [MR03] presents an interesting list of open issues of boosting approaches which must be known by everyone wants to use and develop boosting approaches. The Friedman et al. paper [FHT00], on statistical (probabilistic) boosting, is fundamental to understand the theory under LogitBoost. If one already knows the basic concepts of Boosting, [KSB] is a good tutorial to touch the edge of the research of boosting methods for computer vision problems.

3.2.1.1 Binary Methods

I start with the so-called *AdaBoost*$_{REG}$ [ROM01], where examples that are mislabelled and usually more difficult to classify should be forced to have a positive margin. Assuming that the hard examples are noisy, the algorithm chooses the mistrust parameter at iteration m, $\zeta_n^{(m)}$, an amount by which the example $(\mathbf{x}_n, c_n)$ influenced the decision in the previous iterations.

Another solution is called *BrownBoost* algorithm [Fre01], which is based on the *Boosting by Majority* (BBM) algorithm [Fre95]. An important difference between BBM and AdaBoost is that BMM uses a pre-assigned number of iterations. This strategy leads to the fact that only the examples which have a large margin will eventually be correctly labelled.

SmoothBoost [Ser04] is an intelligent solution to solve the problem of outliers in which the skewness of the data distributions is taken into account. SmoothBoost is similar to AdaBoost in maintaining a set of weights at each iteration, except for the fact that there is a cut-off to the weights assigned to the examples with very negative margin.

The last approach is *MPLBoost* [BDTB08] (Multiple Pose Learning Boosting), which introduces a new discriminative unsupervised clustering procedure embedded into boosting framework. Therefore, the hard examples are split into different categories where these are more easily classifiable. This method is also interesting because it overcomes another limit of AdaBoost and LogitBoost, namely the necessity of aligned visual examples to achieve good classification results. Typically, this operation is done manually by users and it is laborious, but with *MPLBoost* it is possible to learn and align data simultaneously.

A recent interesting boosting approach, termed UBoost [SWSW11], presents a way to exploit the "universum data", i.e. data which belongs to none of the classes of the classification problem of interest, but may contain useful prior domain knowledge to train a classifier. Another recent boosting approach, denoted TaylorBoost, is proposed in [SMSV11]. It supports any combination of loss function and first or second order optimization, and includes classical algorithms, such as AdaBoost, GradientBoost or LogitBoost as special cases.

Finally, it is worth noting that the work described in [ZLSB10], where one can find an on-line semi-supervised learning algorithm, able to combine both boosting and multiple instance learning which has been used to build a tracking-by-detection framework of visual objects.

3.2.1.2 Multi-class Methods

Important approaches for multi-class boosting frameworks share one or both of the following key concepts: first, because of large intra-class and inter-class variation, a *divide-and-conquer* strategy is necessary (for example the tree-structure described in [WN07]). Second, to *share features*, which is an effective and efficient strategy for multi-class learning [TMF07]. Moreover, if the two previous key concepts are integrated with a cascade decision strategy (see [VJV03]), a robust multi-class object detector is built.

In [TMF07], for the first time, it is proposed to share features in a boosting framework termed *JointBoost*. Each feature in any boosting algorithm determines

uniquely a weak classifier, so sharing features means sharing weak classifiers. The underlying idea of [TMF07] is that when detectors are trained jointly, the system looks for features that generalize across multiple classes. Conversely, when detectors are trained independently, the system learns class-specific features. The disadvantage of class-specific features is that for a large number of classes there are not enough computational resources. There is an unclear point in [TMF07], namely how to compute a posterior distribution after a tree classifier is built. This issue is taken seriously into account by [Tu05], where a general boosting-based classification model called *Probabilistic Boosting Tree* (PBT) is proposed. Furthermore, the PBT builds a tree structure recursively from posterior distribution. At each level the input training set is divided into two new sets, in which AdaBoost is applied to make two new strong classifiers.

Vector Boosting, presented in [HALL05], is a multi-class extension of AdaBoost, whose weak learners and final output are vectors rather than scalars. This method produces a multi-class multi-label classifier and is used in [HALL05] to learn branching nodes of a WFS (Width-First-Search) tree structure. The combination of WFS and VectorBoost is adopted to achieve higher performances in both speed and accuracy for multi-class problem. The main limit of the approach is that the tree structure is fixed, so one needs to select sub-categories manually. Multi-class Bhattacharyya boosting (*MBHBoost*) [LL05] avoids the computational burden to build a tree-structured classifier using a single cascade of classifiers where only the most significant information in the training set are considered.

In [ZZMC07, ZPG$^+$06] the multi-class LogitBoost framework is implemented with good results. In [ZPG$^+$06], the multi-class classifier is combined with a tree structure and a cascade structure, but, unlike previous works, the dividing operation at each node is operated at a class level rather than at a sample level. This strategy leads to a lower risk of over-fitting. [ZZMC07] extends [ZPG$^+$06] with a learning procedure called probabilistic boosting network (PBN) for joint real-time object detection and pose estimation. PBN integrates evidence from two building blocks, namely a multi-class boosting classifier for pose estimation as in [ZPG$^+$06] and a boosted detection cascade for object detection. Following the approach in [TMF07], an efficient shared multi-class detection cascade is proposed in [ZMG08], where the detector uses a cascade that joins the handling of similar object classes, separating off classes at appropriate levels of the cascade at the same time.

A recent advancement in multi-class boosting frameworks is represented by [MS11], where a theory of multi-class boosting is presented by making more accurate and identifying the optimal requirements for convergence on the weak classifiers in a multi-class setting. Another interesting work is described in [GB11], in which the authors give the theoretical guarantees necessary for the convergence of smooth convex objective functions with the existing gradient boosting framework [MBB00]. Finally, totally-corrective multi-class boosting is presented in [SH11], that formulates a direct optimization method for training multi-class boosting, unlike most previous multi-class boosting algorithms which decompose a multi-boost problem into multiple independent binary boosting problems.

3.2.2 Binary Logistic Regression Boosting

This Section describes the probabilistic version of boosting family of approaches, which shares the same underlying idea as AdaBoost (see Alg. 1) and that is called LogitBoost (Logistic Regression Boosting). It is proposed in [FHT00]. An adapted version of LogitBoost, able to deal with tensor representation, is used in Chap. 5 for detection problems.

This procedure fits additive logistic regression models by stage-wise optimization of the Bernoulli log-likelihood L that is formulated for a dataset example $\mathbf{x}$ as follows:

$$L = \log(P(c=1|\mathbf{x})) + (1-(c=1))\log(1-P(c=1|\mathbf{x})),$$

where $c \in C = \{0,1\}$ is the label associated with $\mathbf{x}$ and $P(c=1|\mathbf{x})$ is the (posterior) probability to be labelled as 1. It can be written as

$$P(c=1|\mathbf{x}) = P(\mathbf{x}) = \frac{e^{(G(\mathbf{x}))}}{e^{(-G(\mathbf{x}))} + e^{(G(\mathbf{x}))}}.$$

Alg. 2 details binary LogitBoost for two class classification problems, so that $C \in \{-1,1\}$.

Algorithm 2: Binary LogitBoost

Data: A dataset of N couples $\{(\mathbf{x}_1, c_1), \ldots, (\mathbf{x}_N, c_N)\}$, $c_i \in \{0,1\}$, $G(\mathbf{x}_i) = 0$, and probability estimates $p(\mathbf{x}_i) = \frac{1}{2}$.
Result: The classifier G.
begin
  Initialize the observation weights $w_i = 1/N, i = 1, 2, \ldots, N$;
  for $m = 1, 2, \ldots, M$ **do**
    Compute the working responses z_i and weights w_i

$$z_i = \frac{c_i - P(\mathbf{x}_i)}{P(\mathbf{x}_i)(1-P(\mathbf{x}_i))};$$
$$w_i = P(\mathbf{x}_i)(1-P(\mathbf{x}_i);$$

    Fit a the function $f_m(\mathbf{x})$ by a weighted last-square regression of z_i to $\mathbf{x}_i$ using weights w_i;
    Update $G(\mathbf{x}_i) = G(\mathbf{x}_i) + \frac{1}{2} f_m(\mathbf{x}_i)$ and $p(\mathbf{x}_i)$ as

$$P(\mathbf{x}_i) = \frac{e^{(G(\mathbf{x}_i))}}{e^{(G(\mathbf{x}_i))} + e^{(-G(\mathbf{x}_i))}};$$

3.2.3 Multi-class Logistic Regression Boosting

Here LogitBoostis described in its general setting, namely the multi-class case. An adapted version of multi-class LogitBoost, able to deal with tensor representation, is used in Chap. 6 for classification problems.

Let $\{(\mathbf{x}_1, c_1), \ldots, (\mathbf{x}_N, c_N)\}$ be a dataset where $c = \{1, 2, \ldots, K\}$. The log-likelihood function, which generalize the binary one described previously, can be constructed as in [Bis05]

$$L = \sum_{n=1}^{N} \sum_{k=1}^{K} [1\{c_n = k\} \log P(k|\mathbf{x}_n) + (1 - 1\{c_n = k\})(1 - \log P(k|\mathbf{x}_n))], \quad (3.1)$$

where $1\{c_n = k\}$ is an indicator function and $P(k|\mathbf{x}_n)$ is the model probability that $\mathbf{x}_n$ belongs to the k-th class which can be computed as follows.

$$P(k|\mathbf{x}_n) = P_k(\mathbf{x}_n) = \frac{e^{G_k(\mathbf{x}_n)}}{\sum_{j=1}^{K} e^{G_j(\mathbf{x}_n)}}, \quad G_k(\mathbf{x}_n) = \sum_{i=1}^{K} h_k(\mathbf{x}_n), \quad (3.2)$$

in which $e^{G_k(\mathbf{x}_n)} / \sum_{j=1}^{K} e^{G_j(\mathbf{x}_n)}$ is also called *softmax* approximation. The exact form of $h_k(\mathbf{x}_n)$ is

$$h_k(\mathbf{x}_n) = \frac{K-1}{K} \left(f_k(\mathbf{x}_n) - \frac{1}{K} \sum_{j=1}^{K} f_j(\mathbf{x}_n) \right),$$

where $f_k(\mathbf{x}_n)$ is a binary weak hypothesis.

Alg. 3 details multi-class LogitBoost for K class classification problem. In

Algorithm 3: Multi-class LogitBoost

Data: A dataset of N couples $\{(\mathbf{x}_1, c_1), \ldots, (\mathbf{x}_N, c_N)\}$, $c_i \in \{1, 2, \ldots, K\}$, $G(\mathbf{x}_i) = 0$, and probability estimates $P_k(\mathbf{x}_i) = \frac{1}{K}$.
Result: The classifier $G = \{G_1, G_2, \ldots, G_K\}$.
begin
  Initialize the observation weights $w_{ik} = 1/N, i = 1, 2, \ldots, N$;
  for $m = 1, 2, \ldots, M$ **do**
    for $k = 1, 2, \ldots, K$ **do**
      Compute the working responses z_{ik} and weights w_{ik} for the k-th class

$$z_{ik} = \frac{1\{c\}_i - P_k(\mathbf{x}_i)}{P_k(\mathbf{x}_i)(1 - P_k(\mathbf{x}_i))};$$
$$w_{ik} = P_k(\mathbf{x}_i)(1 - P_k(\mathbf{x}_i);$$

      Fit a the function $f_{mk}(\mathbf{x})$ by a weighted last-square regression of z_{ik} to $\mathbf{x}_i$ using weights w_{ik};
      Set $f_{mk}(\mathbf{x}) = \frac{K-1}{K}(f_{mk}(\mathbf{x}) - \frac{1}{K} \sum_{j=1})^K f_{mj}(\mathbf{x})$;
      Set $G_k(\mathbf{x}) = G_k(\mathbf{x}) + f_{mk}(\mathbf{x})$;
      Update $G_k(\mathbf{x}_i) = G_k(\mathbf{x}_i) + \frac{1}{2} f_m(\mathbf{x}_i)$ and $p(\mathbf{x}_i)$ as

$$P_k(\mathbf{x}_n) = \frac{e^{G_k(\mathbf{x}_n)}}{\sum_{j=1}^{K} e^{G_j(\mathbf{x}_n)}};$$

[FHT00] the superior performance of LogitBoost is proved, with respect to the

other generalization obtained from AdaBoost algorithm, including Gentle AdaBoost, AdaBoost.MH and AdaBoost.MR (see [FHT00, MR03] for the details of these algorithms). This is due to the fact that the mentioned competitors of LogitBoost turn multi-class classification problems into a sequence of binary classification problems, so the final classifier is build with a set of separate two-class weak classifiers. Differently LogitBoost deals with multi-class classification problems without split them into sequences of binary classification problems.

The most recent advancement about the theory of multi-class Logitboost is described in [SZ11] with a framework called AOSO-LogitBoost(AdaptiveOne-vs-One LogitBoost). This new LogitBoost behaves as if it combined many one-vs-one binary classifiers adaptively and demonstrates that it leads to higher classification accuracy and faster convergence rate on a number of public datasets.

3.3 Bagging and Random Forests

Before starting to detail the RF method, it is necessary to recall the intuition under Bagging, shared with RF. Bagging is a technique to reduce the variance of an estimated prediction function. Consider first a regression problem. Suppose to fit a model to the training data $\mathcal{Z} = \{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \ldots, (\mathbf{x}_z, c_z)\}$, obtaining the prediction $f(\mathbf{x})$ at input $\mathbf{x}$. Bootstrap aggregation, or bagging, averages this prediction over a collection of bootstrap samples, thereby reducing its variance. For each bootstrap sample $\mathcal{Z}_t$, $t = 1, 2, \ldots, T$, the model is fitted, giving prediction $f_t(\mathbf{x})$. The bagging estimate is defined by

$$f(\mathbf{x}) = \frac{1}{T} \sum_{t=1}^{T} f_t(\mathbf{x}).$$

Bagging seems to work particularly well for high-variance, low-bias procedures, such as trees. As previously shown, for regression or "continuous" classification problems, one can simply fit the same regression tree many times to bootstrap sampled versions of the training data, and average the result. For classification, each committee of trees casts a vote for the predicted class.

RF [Bre01] is a substantial modification of bagging that builds a large collection of decorrelated trees, and then averages them. On many problems the RF performance is very similar to boosting, and they are simpler to train and tune.

3.3.1 State of The Art of Random Forests methods

There is a rising interest on an ensemble of classifiers supervised learning approach called RF [Bre01] for computer vision tasks. RF has demonstrated to be better or at least comparable to other state-of-the-art methods in classification and regression tasks [CNM06]. RF has been applied to keypoint matching [LLF05, MTJ06], segmentation [YCWE07], head pose estimation [FGVG11], human pose detection [SFC$^+$11], object detection and recognition [GYR$^+$11], image classification [BZM07b, MNJ08] and semantic image segmentation [SJC08]. Recently RF has

been combined with several other techniques for several supervised and unsupervised learning tasks. It is worth noting that the most interesting one, which is the combination of RF with Multiple Instance Learning [LSB10], Hough Transform [GYR+11], Conditional Random Field [PT10]. Another interesting reference is a recent generalization of RF [CSK11], termed Decision Forest, used for classification, regression, density estimation, manifold learning and semi-supervised learning problems.

3.3.2 Random Forests

The basic ingredient of a (random) forest is a tree. Trees are ideal candidates for bagging and RF, since they can capture complex interaction structures in the data, and, if grown sufficiently deep, have relatively low bias. Since trees are notoriously noisy, they benefit greatly from the averaging. Moreover, because of the fact that each tree generated in bagging is identically distributed (i.d.), the expectation of the average output is the same as the expectation of any tree. This means the bias of bagged trees is the same as that of the individual trees, and the only prospect to improve is through variance reduction. This is in contrast to boosting, where the weak learners are computed in an adaptive way to remove bias, and hence are not i.d.. An average of T i.i.d. (independent i.d.) random variables, each with variance σ^2, has variance $\frac{1}{T}\sigma^2$. If the variables are simply i.d. with positive pairwise correlation ρ, the variance of the average is

$$\rho\sigma^2 + \frac{1}{T}\sigma^2.$$

As T increases, the second term disappears, but the first remains, hence the size of the correlation of pairs of bagged trees limits the benefits of averaging. The idea in RF (Alg. 4) is to improve the variance reduction of bagging by reducing the correlation between the trees, without increasing the variance too much. This is achieved in the tree-growing process through random selection of the input variables, which, roughly speaking, corresponds to the different dimensions of a feature vector $\mathbf{x}$ (utilized to describe a dataset example). Specifically, when growing a tree on a bootstrapped dataset, before each splits, select $m \leq n$ of the input variables at random as candidates for splitting. Typically, a value for m is $\sqrt{n}$. Intuitively, reducing m the correlation between any pair of trees reduces in the ensemble, and therefore reduces the variance of the average. To make a prediction at a new example represented by $\mathbf{x}$ exploiting a RF learned as in Alg. 4, one should do as follows.

Regression $f(\mathbf{x}) = \frac{1}{T}\sum_{b=1}^{B} T_b(\mathbf{x})$.
Classification Let $c_b(\mathbf{x})$ be the class prediction of the b-th random-forest tree. Then $c = \arg\max_{c\in\{1,2,\ldots,K\}}\{c_b(\mathbf{x})\}_1^T$.

Each tree T_b in a forest is built and tested independently of other trees, hence the overall training and testing procedures can be performed in parallel. During the training, each tree receives a new bootstrapped training set generated by subsampling with replacement of the original training set. I refer to the samples which are not included during the training of a tree as the Out-Of-Bag (OOB) samples of

Algorithm 4: Random Forest

Data: A dataset of z couples $\{(\mathbf{x}_1, c_1), \ldots, (\mathbf{x}_z, c_z)\}$, $c_i \in \{1, 2, \ldots, K\}$.
Result: A Random Forest $\{T_b\}_1^T$.
begin
  for $t = 1, 2, \ldots, T$ **do**
    Draw a bootstrap sample of size n from the training data;
    Grow a random-forest tree T_b to the bootstrapped data, by repeating recursively the following steps for each terminal node of the tree;
    repeat
      Select m variables at random from the n variables;
      Pick the best variable/split-point among the m;
      Split the node into two daughter nodes;
    until $z \geq z_{\min}$;

that tree. These samples can be used to compute the Out-Of-Bag-Error (OOBE) of the tree, in addition to the ensemble which is a low-biased estimate of the generalization error. An OOBE estimate is almost identical to the one obtained by K-fold cross validation. Hence, unlike many other non-linear estimators, RF can be fit in one sequence, with cross-validation being performed along the way. Therefore, a standardized stop criterion to terminate automatically the training phase is to look at the OOBE and, once it stabilizes the training, it can be terminated.

RF also uses the OOB samples to construct a different *variable importance* measure (similarly to boosting), to calculate the prediction strength of each variable. When the b-th tree is grown, the OOB samples are passed down the tree, and the prediction accuracy is recorded. Then the values for the j-th variable are randomly permuted in the OOB samples, and the accuracy is computed again. The decrease in accuracy, as a result of this permuting, is averaged over all trees, and is used as a measure for the importance of the variable j in the random forest.

Finally, before analysing the Kernel Methods, a simple comparison between RF and LogitBoost is made, which can be useful to choose the right model in practice. For what concerns RF: (1) it has good generalization performances, (2) it has very fast learning and testing phases, (3) it is inherently multi-class, and (4) it has few parameters for the training phase. Unfortunately, the learning of different RF on the same data could be not consistent and it is not adaptive as Boosting because different iterations of the algorithm, during the training phase, are not tweaked in favour of the instances misclassified by previous iterations. For what concerns LogitBoost,(1) it has a much stronger theoretical background if compared with RF, because its consistency is proven, (2) it guarantees good generalization performances, (3) it is characterized by a fast training and testing phase, even if RF is faster in the training phase.

3.4 Kernel Methods

For what concerns the kernels methods and in particular SVMs with kernels, they are a way to learn and apply discriminative models efficiently in very high dimen-

sional (such as infinite-dimensional) feature spaces or/and dealing with non-vector (i.e. tensor) inputs. Before detailing how to build a kernel and learn from it exploiting SVMs, I briefly give the idea under SVMs using the introduction given by Andew Ng [Ng07]. To learn more about kernel methods and SVMs, good books are [CST04, Bel06, HTF11, Pri12].

SVMs are built on the concept of (geometric) margin that can be considered as a sort of "confidence" of the predictions made by an SVM classifier. Given a training set, the goal is to try to find a decision boundary that maximizes the margin, since this would reflect very confident predictions on the training set and therefore a good "fit" of the discriminative model to the training data. Specifically, this will result in a classifier that separates the positive and the negative training examples with a "gap" (margin). Assuming to be given a training set that is linearly separable (i.e. that it is possible to separate the positive and negative examples using some separating hyperplane), to find the maximum (or optimal) margin one should solve the following optimization problem:

$$\max_{\gamma,\mathbf{w},\mathbf{b}} \gamma, \quad \text{s.t. } c_i(\mathbf{w}^T\mathbf{x} + \mathbf{b}) \geq \gamma, \quad i = 1, \ldots, m.$$

This corresponds to maximize the margin γ subject to each training example having margin at least γ. $(\mathbf{w}, \mathbf{b})$ are the parameters that govern the separation hyperplane. To the above-mentioned formulation, in order to find the optimal solution, the constrain $||\mathbf{w}|| = 1$ should be imposed. But this constraint is a non-convex one, so the optimization problem certainly is not in any format that can be plugged into standard optimization software to solve. However one can add an arbitrary scaling constraint on $\mathbf{w}$ and $\mathbf{b}$ without changing anything. So, after some math, one can reformulate the previous optimization problem as

$$\min_{\gamma,\mathbf{w},\mathbf{b}} \frac{1}{2}||\mathbf{w}||^2, \quad \text{s.t.} c_i(\mathbf{w}^T\mathbf{x} + \mathbf{b}) \geq 1, \quad i = 1, \ldots, m.$$

Now the problem is transformed into a form that can be efficiently solved. The above-mentioned problem is an optimization one, with a convex quadratic objective and only linear constraints. Its solution gives the optimal margin classifier. This optimization problem can be solved using the commercial quadratic programming (QP) code.

Since Kernel Methods are a family of standardized methods I refer to [CST04, Bel06, HTF11, Pri12] for a good state of the art.

3.4.1 Fundamental Concepts of Kernel Methods

Many linear models for regression and classification (e.g. linear regression) can be reformulated into an equivalent *dual representation* in which also the predictions are based on linear combinations of a *kernel* function evaluated at the data. For models based on a fixed non-linear feature space mapping $\phi(\mathbf{x})$, the kernel function is given by the relation

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j).$$

From this definition, one can see that the kernel is a symmetric function of its arguments such that $k(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_j, \mathbf{x}_i)$. Intuitively, one can consider the function

k as a similarity measure among examples. Obviously, in general, given complex examples, a non-linear measure should be used to model that complexity. Then, kernel methods represent an example in the dataset as the collection of similarities between that example and all the other examples in the dataset. Hence, rather than applying SVMs using the original input attributes $\mathbf{x}$, my purpose is to learn using some features $\phi(\mathbf{x}_i)$. Since the SVM optimization problem can be written entirely in terms of inner products $\mathbf{x}_i^T\mathbf{x}_j$ this means that one can replace all those inner products with $\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$. Therefore, given ϕ, one can easily compute $k(\mathbf{x}_i, \mathbf{x}_j)$ by finding $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ and taking their inner product. But what is more interesting is that $k(\mathbf{x}_i, \mathbf{x}_j)$ is often relatively inexpensive to calculate.

Consider some finite sets of m points $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$, and let a square, $m \times m$ matrix K be defined so that its (i, j)-entry is given by $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. This matrix is called the *Kernel matrix*. If K is a valid Kernel, which imply a fast convergence of the learning (optimization) procedure, then

$$K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j) = \phi(\mathbf{x}_j)^T\phi(\mathbf{x}_i) = k(\mathbf{x}_j, \mathbf{x}_i) = K_{ji},$$

hence K must be symmetric. Moreover, let $\phi_k(\mathbf{x})$ denote the k-th coordinate of the vector $\phi(\mathbf{x})$, so that one find for any vector $\mathbf{z}$ the following condition

$$\begin{aligned}
\mathbf{z}^T K\mathbf{z} &= \textstyle\sum_i \sum_j \mathbf{z}_i K_{ij} \mathbf{z}_j \\
&= \textstyle\sum_i \sum_j \mathbf{z}_i \phi(\mathbf{x}_j)^T \phi(\mathbf{x}_i) \mathbf{z}_j \\
&= \textstyle\sum_i \sum_j \mathbf{z}_i \sum_k \phi_k(\mathbf{x}_j)^T \phi_k(\mathbf{x}_i) \mathbf{z}_j \\
&= \textstyle\sum_k \sum_i \sum_j \mathbf{z}_i \phi_k(\mathbf{x}_j)^T \phi_k(\mathbf{x}_i) \mathbf{z}_j \\
&= \textstyle\sum_k \sum_i \left(\mathbf{z}_i \phi_k(\mathbf{x}_i)\right)^2 \\
&\geq 0.
\end{aligned}$$

Hence, if K is symmetric positive semi-definite. This condition for K it necessary and sufficient to be a valid kernel (also called a Mercer kernel). The following result is due to Mercer.

Theorem 3.1 (Mercer). *Let $K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ be given. Then for K to be a valid (Mercer) kernel, it is necessary and sufficient that for any $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$, $(m < \infty)$, the corresponding kernel matrix is symmetric positive semi-definite.*

All the ingredients of a kernel machine have been presented, and the four key aspects characterizing a kernel method can be presented using the schematic in Alg. 5

3.4.2 Design Kernels from Tensor Metrics

Regarding the design of kernel for covariance matrices, the following procedure is adopted. Given a dataset $\{\mathbf{X}_i, y_i\}_{i=1,\ldots,N}$ where $\mathbf{X}_i$ are tensors and y_i the associated labels

1. Choose a distance between a pair of tensors $d(\mathbf{X}_i, \mathbf{X}_j)$ (See Sec. 3.5.2 to get confident about tensor distances).

Algorithm 5: Kernel Machine

Data: A dataset of m couples $\{(\mathbf{x}_1, c_1), \ldots, (\mathbf{x}_m, c_m)\}$, $c_i \in \{1, 2, \ldots, K\}$.
Result: A Kernel Machine.
begin
$\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ are embedded into a vector space, called the feature space, using a kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$;
Linear relations are sought among the images of the data items in the feature space and stored in the kernel matrix $K(\mathbf{x}_i, \mathbf{x}_j)$;
The learning algorithms (i.e. SVM) are implemented in such a way that the coordinates of the embedded points are not needed, only their pairwise inner products;
The pairwise inner products can be computed efficiently directly from the original data items using $k(\mathbf{x}_i, \mathbf{x}_j)$;

2. Build a distance (dissimilarity) matrix $D(\mathbf{X}_i, \mathbf{X}_j)$ as

$$D(\mathbf{X}_i, \mathbf{X}_j) = \begin{bmatrix} d(\mathbf{X}_1, \mathbf{X}_1) & \cdots & d(\mathbf{X}_n, \mathbf{X}_1) \\ \vdots & \ddots & \vdots \\ d(\mathbf{X}_1, \mathbf{X}_n) & \cdots & d(\mathbf{X}_n, \mathbf{X}_n) \end{bmatrix}.$$

3. Turn D into a kernel (similarity) matrix as

$$K = \exp\left(\frac{-1}{\mu(D)} D\right)$$

where $-1/\mu(D)$ is a regularization term in which $\mu(D)$ is the average value of D.

Since valid kernels are symmetric and positive semi-definite, it is not possible to use directly those distances to build a valid kernel matrix. Hence, I apply a simple and effective mapping which permits to turn a distance (dissimilarity) matrix into a kernel matrix. A distance matrix D is made by the distance among all pairs of training covariance matrices, therefore it is a symmetric matrix. It can be turned into a similarity matrix applying the above-mentioned (non-linear) exponential transformation of its entries.

3.4.3 Learning from Tensors with Kernel Methods

A kernel machine can be built using the procedure as follows.

1. Choose a kernel function $k(\mathbf{X}_i, \mathbf{X}_j)$, where $\mathbf{X}_i, \mathbf{X}_j$ are two tensors.
2. Define kernel matrix (also known as *Gram matrix*) $K(\mathbf{X}_i, \mathbf{X}_j)$ as

$$\begin{bmatrix} k(\mathbf{X}_1, \mathbf{X}_1) & \cdots & k(\mathbf{X}_n, \mathbf{X}_1) \\ \vdots & \ddots & \vdots \\ k(\mathbf{X}_1, \mathbf{X}_n) & \cdots & k(\mathbf{X}_n, \mathbf{X}_n) \end{bmatrix},$$

3. Choose and apply a machine learning algorithm (i.e. an SVM) to learn the parameters $\{\alpha_i\}_{i=1,\dots,n}$ of model f.
4. Apply the model to a new tensor $\mathbf{X}_{n+1}$:

$$f(\mathbf{X}_{n+1}) \sum_{i=1}^{n} \alpha_i k(\mathbf{X}_i, \mathbf{X}_{n+1}).$$

where n is the number of examples in your dataset.

The most interesting characteristics of that procedure are: (1) the fact that exploiting their dual representation they only need inner products between dataset examples (not their coordinates), therefore kernel machine is a perfect tool to manage tensor representation; (2) their modularity. In fact, using this family of methods it is possible to separate the design of the machine learning algorithm from the design of the kernel matrix.

3.5 Cases of Interest

3.5.1 A Vector Classification Framework for Sym_d^+

Alg. 6 describes a generic training framework for multi-class classification problems using SPD tensors. It can use whatever binary or multi-class vector supervised classification algorithm (i.e. LogitBoost and RFs) and in this thesis is use, in Sec. 5.3,5.6,6.3,6.5, and 6.6. The idea under this simple generalization is to linearise Sym_d^+ which is actually non-flat "hoping" that the distortion introduced by the approximation is low. Hence, Alg. 6 is a valid procedure to learn a discriminative model on Sym_d^+. Now, it is simple to specialize this procedure (Alg. 6) for RF

Algorithm 6: A Vector Classification Framework for Sym_d^+

Data: $(\mathbf{X}_1, y_1), \dots, (\mathbf{X}_n, y_n)$ with $\mathbf{X}_i \in Sym_d^+$ and $y_i \in \{1, \dots, K\}$ the class label.
Result: The classifier C.
begin
Linearise Sym_d^+ computing the standard log of a matrix $\xi_{\mathbf{X}_i} = \log(\mathbf{X}_i)$;
Vectorize $\xi_{\mathbf{X}_i}$ as $\xi_{\mathbf{x}_i} (\in \mathbb{R}^d) = \text{vec}(\xi_{\mathbf{X}_i})$ as described on Sec.2.2.6;
Learn a classifier $C(\xi_{\mathbf{x}_i}) : \mathbb{R}^d \mapsto \{1, \dots, K\}$ using any standard (vector) supervised learning technique (RFs, Boosting, etc.);

(Alg. 4) and LogitBoost (Alg. 2 and Alg. 3). It is necessary to highlight that in Alg. 6 the log operator in the standard logarithm of a matrix one.

3.5.2 Kernel Frameworks

3.5.2.1 Sym_d^+

Alg. 7 describes a simple learning procedure which trains a kernel machine on Sym_d^+ which exploits a distance measure among SPD matrices which is used in

Sec. 4.3,5.4,6.4,and 6.7. I refer to the training set by $\{\mathbf{X}_i, y_i\}_{i=1,\dots,n}$, where $\mathbf{X}_i$ are SPD matrices. Alg. 7 exploits introduces an approximation of Sym_d^+ and it works

Algorithm 7: Kernel Methods on Sym_d^+

Data: $(\mathbf{X}_1, y_1), \dots, (\mathbf{X}_n, y_n)$ with $\mathbf{X}_i \in Sym_d^+$ and $y_i \in \{1, \dots, K\}$ the class label.
Result: K and the SVM parameters.
begin
Build a Distance Matrix $D(\mathbf{X}_i, \mathbf{X}_j) = \mathrm{tr}((\log(\mathbf{X}_i) - \log(\mathbf{X}_j))^2)$;
Build a Kernel Matrix $K(\xi_{\mathbf{X}_i}, \xi_{\mathbf{X}_j})$ as described in Sec. 3.4.2;
The learning algorithms (i.e. SVM) are implemented in such a way that the coordinates of the embedded points are not needed, only their pairwise inner products;

well only if the curvature of Sym_d^+ is low. To adopt more sophisticated distances among SPD tensors, refer to Sec. 6.4.

3.5.2.2 $Grass(p, n)$

As shown in Sec. 2.6.2, a point $\mathbf{X} \in Grass(p, n)$ is a (p-dimensional) subspace of $\mathbb{R}^n$. One can easily build a "collection" of p n-dimensional vectors, but may ask what is possible to describe with a collection of vectors. The most representative examples are action classification [TVC08, LBK10] and object categorization [LYT06, HL08, TVC08, HL09, CV09]. $Grass(p, n)$ math for computer vision and machine learning problems is well explained in [TVC08, HL08, SM09].

Given one of the distances between two subspaces $\mathbf{X}, \mathbf{Y}$ of $\mathbb{R}^n$ described in [TVC08], for example

$$d(\mathbf{X}, \mathbf{Y}) = (n - \sum_{i=1}^{n} \cos^2 \theta_i)^{1/2}, \tag{3.3}$$

where θ_i is a singular value (see Sec. 2.2.5) of $\mathbf{X}'\mathbf{Y}$, I can easily derive a kernel for $Grass(p, n)$, exploiting the procedure described in Sec. 3.4.2. Therefore, similarly to the previous case, Alg. 8 describes how to learn a kernel machine on $Grass(p, n)$

3.5.2.3 Hausdorff Spaces

Finally, I show the procedure to learn a kernel machine on a tensor space in which the Hausdorff distance is used. Even though the procedure to build a kernel machine on a tensor space is surely clear, at this point, it is worth noting this case for two reasons: first, this method is used in Sec.5.4; second, it is a procedure that can be handled as a flexible object description. To be more precise, given two sets of vectors $\mathbf{X}$ and $\mathbf{Y}$, they do not need to contain the same number of vectors. Therefore, if $\mathbf{X} \in \mathbb{R}^m$ and $\mathbf{Y} \in \mathbb{R}^n$, then m and n can be different. The Hausdorff distance is formulated as:

Algorithm 8: Kernel Methods on $Grass(p, n)$

Data: $(\mathbf{X}_1, y_1), \ldots, (\mathbf{X}_n, y_n)$ with $\mathbf{X}_i \in Grass(p, n)$ and $y_i \in \{1, \ldots, K\}$ the class label.
Result: K and the SVM parameters.
begin
Build a Distance Matrix $D(\mathbf{X}_i, \mathbf{X}_j) = (n - \sum_{i=1}^{n} \cos^2 \theta_i)^{1/2}$;
Build a Kernel Matrix $K(\mathbf{X}_i, \mathbf{X}_j)$ from $D(\mathbf{X}_i, \mathbf{X}_j)$ as described in Sec. 3.4.2;
The learning algorithms (i.e. SVM) are implemented in such a way that the coordinates of the embedded points are not needed, unlike their pairwise inner products;

$$d(\mathbf{X}, \mathbf{Y}) = \max[\max_{\mathbf{x} \in \mathbf{X}}(\min_{\mathbf{y} \in \mathbf{Y}}(||\mathbf{x}, \mathbf{y}||)), \max_{\mathbf{y} \in \mathbf{Y}}(\min_{\mathbf{x} \in \mathbf{X}}(||\mathbf{y}, \mathbf{x}||))] \quad \mathbf{x}, \mathbf{y} \in \mathbb{R}^z, \tag{3.4}$$

and it has been used in object classification and detection [HKR93, SKP99, JKF01]. Alg. 9 describes how to learn a kernel machine on Hausdorff spaces.

Algorithm 9: Kernel Methods on Hausdorff Spaces

Data: $(\mathbf{X}_1, y_1), \ldots, (\mathbf{X}_n, y_n)$ where $\mathbf{X}_i$ is a set of vectors in $\mathbb{R}^z$ and $y_i \in \{1, \ldots, K\}$ the class label.
Result: K and the SVM parameters.
begin
Build a Distance Matrix
$D(\mathbf{X}_i, \mathbf{X}_j) = \max[\max_{\mathbf{x} \in \mathbf{X}_i}(\min_{\mathbf{y} \in \mathbf{X}_j}(||\mathbf{x}, \mathbf{y}||)), \max_{\mathbf{y} \in \mathbf{X}_j}(\min_{\mathbf{x} \in \mathbf{X}_i}(||\mathbf{y}, \mathbf{x}||))]$;
Build a Kernel Matrix $K(\xi_{\mathbf{X}_i}, \xi_{\mathbf{X}_j})$ from $D(\mathbf{X}_i, \mathbf{X}_j)$ as described in Sec. 3.4.2;
The learning algorithms (i.e. SVM) are implemented in such a way that the coordinates of the embedded points are not needed, unlike their pairwise inner products;

4

Tensor Representation for Object Description

Contents

4.1 Introduction

A key problem in object recognition is finding a suitable object representation (or description). For historical and computational reasons, vector descriptions that encode particular statistical properties of the data have been broadly applied. However, by employing tensor (matrix) representation, one is able to describe the interactions of multiple factors inherent to image formation. A successful work, which inspired mine, is the covariance matrix [TPM06, TPM08], briefly described in Sec. 4.2.1, that has demonstrated to lead to state-of-the-art results for several classification and detection tasks. More generally, structure and deformation tensors are used in image understanding, especially for segmentation, grouping, motion analysis and texture segmentation [BWBM06], and can also be utilized in regularization approaches for medical image registration [ACW$^+$07, FPAA07]. My goal is to understand if one can exploit the tensor representation to build a more powerful object descriptor with respect to the covariance, combining different sources of information to obtain better classification and detection accuracy results.

In this Chapter, novel kinds of tensor representation are proposed. Sec. 4.2.2 introduces EMI (Entropy and Mutual Information), a tensor composed of mixing entropy and mutual information, which shows its potentiality in visual object classification problems, where it outperforms covariance representation. Sec. 4.2.3 presents SST (Self Similarity Tensor), which measures the self-similarity of object parts or image features similarly to COV and EMI. Finally, Sec. 4.2.4 describes the Grassmann tensor, that, unlike the previous matrix tensors, represents an object in the form of a *set of feature vectors*.

For what concerns the EMI tensor, it has a Sym_d structure and, for each source of information, it uses histogram as intermediate representation. Then, the entropy and mutual information measures are computed from the histograms to populate the entries of the tensor. EMI is applied to general object classification problems and finer human body part classifications, discovering that EMI tensor leads to considerably better performances than the COV representation, even if its computational cost is higher if multiple instances of EMI are utilized to describe an object, due to the usage of histograms[1].

The SST has been used in two different settings: first, using a robust regular grid structure [DT05, LSP06, TFC+10] and a single source information coming from all the patches, the structural information of an object is characterized. Second, exploiting multiple sources of information and comparing each other, the content of an image is described. I expect that, utilizing the first setting, SST should be better suited for the detection task; while, employing the second setting, SST should be more appropriate for classification purposes.

Regarding the Grassmann tensor, similarly to the structural SST, it is used to characterise the structure of an object, but it uses a set of vectors instead of a matrix representation. It has a fundamental advantage if compared to matrix tensors: the possibility to represent an object with a variable number of vectors. This feature can be useful in case of occlusions or crowded scenarios.

Finally, in Sec. 4.3 an experimental study on object representation using the tensor description is presented.

4.2 Tensor Representations

4.2.1 Covariance Tensors

In this Section, I briefly recap how to build a covariance matrix for classification and detection purposes as depicted in Fig. 4.1, which is well described in [TPM06].

Mathematically speaking, a covariance tensor corresponds to an SPD (Symmetric Positive Definite) matrix and the value of its determinant is a direct measure of the dispersion of the associated Gaussian multivariate random variable.

Given an image I of the image, one can compute the covariance tensor COV of d image features

$$\Phi(I) = [F_1(I),\ F_2(I), \ldots,\ F_d(I)]$$

[1] EMI cannot exploit the integral histogram representation proposed in [Por05]. This is because it does not provide a way to compute the joint histogram which is necessary for EMI tensor.

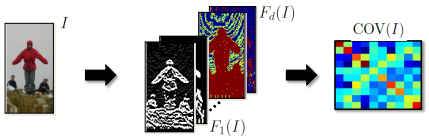


Fig. 4.1. COV descriptor. The d-dimensional feature map set $\Phi(I)$ is constructed from input image I.

such as color, image derivatives, etc., according to:

$$\mathrm{COV}_{ij}(I) = \frac{1}{|I|-1}\sum_{p\in I}(F_i(p)-\mu(F_i))(F_j(p)-\mu(F_j))^T, \quad i,j\in\{1,\ldots,d\}. \tag{4.1}$$

where μ is the mean operator and $|\cdot|$ denotes the set size operator. To visualize a covariance matrix, one can act in the following way. A COV matrix is a Sym_d^+ matrix where the diagonal elements encode the variance, while the off-diagonal elements cipher the correlation between each pair of image features in $\Phi(I)$. Therefore a COV matrix is defined as follows:

$$\mathrm{COV}(I) = \begin{bmatrix} \mathrm{var}(F_1(I),F_1(I)) & \cdots & \mathrm{corr}(F_1(I),F_d(I)) \\ \vdots & \ddots & \vdots \\ \mathrm{corr}(F_d(I),F_1(I)) & \cdots & \mathrm{var}(F_d(I),F_d(I)) \end{bmatrix}. \tag{4.2}$$

The covariance matrix is a very informative descriptor, in fact it encodes the spatial layout of the features and their variance and correlation. Exploiting the fact that covariance coefficients can be expressed in terms of first and second order integral images [TPM06, TPM08] the computation of a covariance matrix can cost only $O(d^2)$ operations.

4.2.2 Entropy-Mutual Information Tensor

Similarly to covariance matrices, EMI tensor is a dense region descriptor. In fact, given an image I of $W\times H$ pixels and a set of d feature maps $\Phi(I)$ of $W\times H\times d$ pixels:

$$\Phi(I) = [F_{1_{W\times H}}(I)),\ F_{2_{W\times H}}(I),\ldots,F_{d_{W\times H}}(I)], \tag{4.3}$$

where $F_1,\ldots,F_d$ are the image features. Then, one can use $\Phi(I)$ to build d histograms of n bins:

$$H(\Phi(I)) = [h(F_1(I))_{1\times n}, h(F_2(I))_{1\times n},\ldots,h(F_d(I))_{1\times n}], \tag{4.4}$$

in which h is the operator used to build a histogram. In order to obtain a probability distribution from each feature, it is necessary to normalize each row of $H(\Phi(I))$, such as $\sum_{i=1}^{n} h(F_j(I))_n = 1$ and $j\in\{1,\ldots,d\}$. The normalized version of $H(\Phi(I))$ is denoted as $\hat{H}(\Phi(I))$:

$$\hat{H}(\Phi(I)) = [\hat{h}(F_1(I))_{1\times n}, \hat{h}(F_2(I))_{1\times n},\ldots,\hat{h}(F_d(I))_{1\times n}]. \tag{4.5}$$

Using Eq. (4.5), the EMI tensor is defined as:

$$\mathrm{EMI}(I) = \begin{bmatrix} \mathrm{E}(\hat{H}_1(\Phi(I))) & \cdots & \mathrm{MI}(\hat{H}_{1d}(\Phi(I))) \\ \vdots & \ddots & \vdots \\ \mathrm{MI}(\hat{H}_{d1}(\Phi(I))) & \cdots & \mathrm{E}(\hat{H}_d(\Phi(I))) \end{bmatrix}, \tag{4.6}$$

where $\mathrm{E}(\hat{H}_i(\Phi(I)))$ is the entropy operator defined as

$$\mathrm{E}(\hat{H}_i(\Phi(I))) = \sum_{j=1}^{n} \hat{h}(F_i(I))_j \log(\hat{h}(F_i(I))_j) \quad i \in \{1, \ldots, d\}, \tag{4.7}$$

and $\mathrm{MI}(\hat{H}_{d1}(\Phi(I)))$ is the mutual-information operator

$$\mathrm{MI}(\hat{H}_{ij}(\Phi(I))) = \sum_{l=1}^{n}\sum_{k=1}^{n} \hat{h}(F_i, F_j(I))_{lk} \log(\frac{\hat{h}(F_i, F_j(I))_{lk}}{\hat{h}(F_i(I))_l \hat{h}(F_j(I))_k}) \quad i, j \in \{1, \ldots, d\}. \tag{4.8}$$

The joint probability is represented in Eq. (4.8) as $\hat{h}(F_i, F_j(I))$. As previousle mentioned, EMI matrix belongs to Sym_d of real numbers. For classification purposes, a minimal representation EMI is defined. Since it has only $d(d+1)/2$ independent coefficients, which are the upper triangular or lower triangular part of the matrix, I decide to consider only the upper triangular part and vectorize it. The resulting vector belongs to $\mathbb{R}^{\frac{d(d+1)}{2}}$ and the standard machine learning framework can be used with this representation.

4.2.3 Self-Similarity Tensor

This Section investigates a tensor called *Self-Similarity Tensor* (SST) which can be used to describe an object robustly both for classification and detection purposes. SST is similar in spirit to structure tensors, which are powerful tools that can be used in such computer vision tasks as edge or corner detection [Tri04], spatio-temporal recognition [LCSL07] and the similarity-based pattern recognition approaches [BMF04, CCFM08, BP09]. The concept of self-similarity is explored in two ways: first, considering an object as composed by parts, the distance (or alternatively similarity) among different parts represented by the same (unique) feature description is compared. This to obtain a sort of *structural characterization* of an object. Second, combining the ideas below the object bank representation [LSXFF10] and the dual representation of kernel methods (see Sec. 3.4), the distance of some image features is compared to provide a sort of *content characterization* of an object.

From a mathematical point of view, in the first case (structural characterization) the intuition is that, given a patch-based representation of an object, it can be possible to find a compact and useful object description capturing the relationship between patches: $\mathrm{SST}_{\mathrm{struct}}$. It is a Sym matrix of distances among the patches (or parts). Then, $\mathrm{SST}_{\mathrm{struct}}$ can be vectorized and used as an object descriptor. More precisely, given an image I of $W \times H$ pixels and a set $\Lambda(I)$ of $W \times H \times m$ pixels of

m image patches described by any kind of feature description (like HOG [DT05], COV [TPM06], LBP [MYL+08], etc.):

$$\Lambda(I) = [f_{1\times n}(P_1(I)),\ f_{1\times n}(P_2(I)), \ldots, f_{1\times n}(P_m(I))], \tag{4.9}$$

where f is a function which produces an n-dimensional vector descriptor (see Eq.(2.1)) and P extracts a patch from the image I. Using Eq. (4.9), the $\mathrm{SST}_{\mathrm{struct}}$ is defined as:

$$\mathrm{SST}_{\mathrm{struct}}(I) = \begin{bmatrix} d(f(P_1(I)), f(P_1(I))) & \cdots & d(f(P_1(I)), f(P_m(I))) \\ \vdots & \ddots & \vdots \\ d(f(P_m(I)), f(P_1(I))) & \cdots & d(f(P_m(I)), f(P_m(I))) \end{bmatrix}. \tag{4.10}$$

For what concerns the second case (content characterization), the idea is similar to COV tensor (see Sec.4.2.1), but in this case a vector distance (like the Euclidean distance) is utilized to measure the "difference" between image features (like color, image derivatives, etc.), so a Sym matrix of distances can be built. It is called $\mathrm{SST}_{\mathrm{content}}$ tensor. Then, $\mathrm{SST}_{\mathrm{content}}$ can be vectorized and used as an object descriptor. More precisely, given an image I of $W \times H$ pixels and a set $\Phi(I)$ of $W \times H \times d$ image features (like color, directional derivatives, etc.):

$$\Phi(I) = [f(F_{1_{W\times H}}(I))),\ f(F_{2_{W\times H}}(I)), \ldots, f(F_{d_{W\times H}}(I))], \tag{4.11}$$

where f is a function which produces an n-dimensional vector descriptor (see Eq.(2.1)) and F extracts a feature from the image I. Using Eq. (4.11), the $\mathrm{SST}_{\mathrm{content}}$ is defined in the following way:

$$\mathrm{SST}_{\mathrm{content}}(I) = \begin{bmatrix} d(f(F_1(I)), f(F_1(I))) & \cdots & d(f(F_1(I)), f(F_d(I))) \\ \vdots & \ddots & \vdots \\ d(f(F_d(I)), f(F_1(I))) & \cdots & d(f(F_d(I)), f(F_d(I))) \end{bmatrix}, \tag{4.12}$$

where d represents any distance function for a pair of n-dimensional vectors.

4.2.4 Grassmann Tensor

This Section describes how to exploit a different tensor representation in the form of a *set of feature vectors*. Each set can be represented as a point of a Grassmann manifold for which the learning schematic defined in Alg. 8 is adopted. Typically this tensor is used in the following scenarios: action classification [TVC08, LBK10] and object categorization [LYT06, HL08, TVC08, HL09, CV09]. The idea, in the latter case, is to represent a visual object from multiple pictures of the individual, taken from different angles, under different illumination or at different places. When comparing such sets of image vectors, one can define the similarity between sets, considering those sets as linear submanifolds of a Euclidean space (see Sec. 2.4.1 for details). Therefore, the problem of learning from submanifolds is formulated on a Grassmann manifold.

Due to the fact that only one image is available for an object, recalling the idea under $\mathrm{SST}_{\mathrm{struct}}$, an object is represented as a set of parts (i.e. patches) of

an image I. Hence, the set of vectors $\Lambda(I)$ (4.9) is exactly the object descriptor. According to the experiments, I found that the best choice to represent each single feature vector is to adopt the LBP [OPM02] features which perform much better if compared to HOG features. This is because the adopted datasets contain low resolution objects and, in this condition, LBP has demonstrated to lead to state-of-the-art performances.

It must be highlighted that, to obtain the best performances from the Grassmann tensor (GRT), its normalised version must be computed. Given a pair of submanifolds $\mathbf{X}$ and $\mathbf{Y}$, one has to compute the matrix product $\mathbf{Z} = \mathbf{X}\mathbf{Y}'$ first, to compute the distance between them. Therefore, the idea is to replace $\mathbf{Z}$ with its normalised version $\hat{\mathbf{Z}}$ that can be computed as:

$$\hat{\mathbf{Z}} = \operatorname{diag}\left(\frac{1}{\sqrt{\mathbf{Z}}}\right)\mathbf{Z}\operatorname{diag}\left(\frac{1}{\sqrt{\mathbf{Z}}}\right), \tag{4.13}$$

where $\operatorname{diag}(\mathbf{M})$ is equal to $\mathbf{M}$, at the diagonal entries, and the rest is truncated to zero. The same normalization is used to compute normalized kernel matrices [CST04] and a similar one is utilized for COV tensors [TPM08]. Then, to compute the distance between $\mathbf{X}$ and $\mathbf{Y}$ using $\hat{\mathbf{Z}}$, the projection distance (3.3) is adopted.

Besides, the set of vector representation has a fundamental advantage than the matrix one, that is the possibility to represent an object with a variable number of vectors that can be useful in case of occlusions or crowded scenarios. On the contrary, the main drawback is represented by its computational cost that is higher if compared to the previous matrix tensor one. This is because the distance between a pair of (Grassmann) tensors needs the usage of the SVD decomposition (see Sec. 2.2.5). It must be highlighted that GRT performs poorly if it is exploited to characterise the content of an image using different image features. This is due to the fact that the sub-manifold, built by the vectorised image features, is meaningless.

4.3 An Experimental Study on Tensor Representation

In this experimental section, a kernel SVM is learned as described in Alg. 7 and LibLinear [FCH+08], in order to compare COV, EMI, $\text{SST}_{\text{content}}$, $\text{SST}_{\text{struct}}$, and GRT tensors. For all of these, in the training phase an 8-fold cross-validation strategy is adopted. For what concerns the learning parameter C, the grid-search is used varying it in $2^{-3}, \ldots, 2$ with step 1.

Owing to the fact that some tensors (i.e. COV, EMI, and $\text{SST}_{\text{content}}$) and the others ($\text{SST}_{\text{struct}}$, GRT) are devoted to characterize the content of an object and the structure respectively, the results keep these two classes separated. At the end of this experimental section, one should be able to decide which is the best way to represent the content and the structure of an object, exploiting the tensor representation. This could be useful to choose the best tensor and to combine tensor representations with different functions.

With regard to the adopted feature, for COV, EMI, and $\text{SST}_{\text{content}}$ tensors, given an image I in the dataset, a set $\Phi(I)$ of d features, where $d = 13$ and x, y are the pixel location, is composed of:

$$\Phi(I,x,y) = \left[F_1(Y)\dots F_8(Y)\ Y\ C_b\ C_r\ G_{|\,|}(Y)\ G_O(Y)\right], \tag{4.14}$$

where $F_1(Y)\dots F_8(Y)$ is the filter bank, consisting of scaled symmetric DOOG (Difference Of Offset Gaussian) [Dol], applied only on the luminance channel of the perceptually uniform CIELab color space. Y, C_b and C_r are the three color channels obtained by transforming the original RGB image. $G_{|\,|}(Y)$ and $G_O(Y)$ are the gradient magnitude and orientation calculated on the Y channel map, respectively.

On the contrary, $\mathrm{SST}_{\mathrm{struct}}$ and GRT are based on the LBP features [VF08] with a patch dimension of 16×16 pixels. As mentioned in Sec.4.2.4, LBP represents the best choice for tiny or low resolution objects.

In the following experiments, two main types of classification tasks for the experimental comparison of the tensors are considered: (1) human characterization tasks, such as body or head orientation estimation (Sec. 4.3.1,4.3.2, 4.3.3, and 4.3.4); and (2) object classification (Sec. 4.3.5).

Finally, it is worth highlighting that different tensors lie in different spaces, i.e. Sym^+ and to Sym, so different metrics are adopted. In particular, for COV tensors the geodesic CBH distance (see Sec. 6.4 where is treated in greater detail) is adopted.

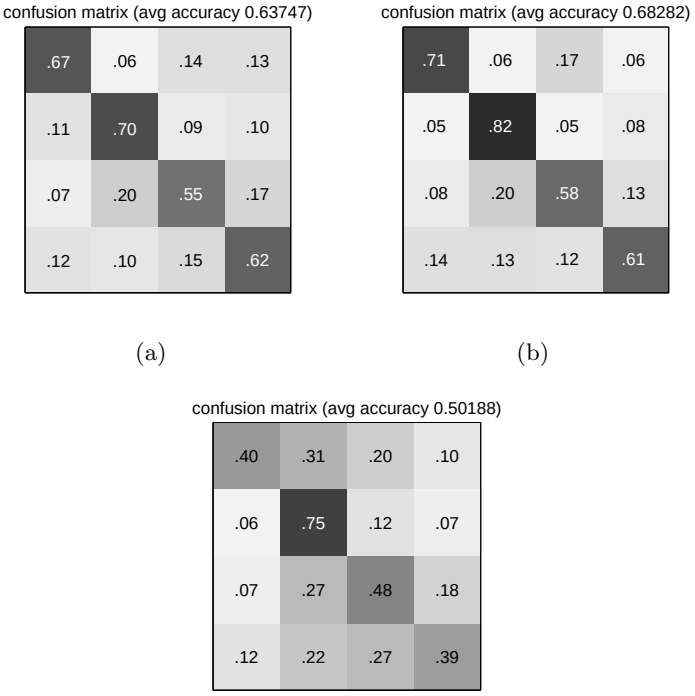
4.3.1 HOC Human Dataset

For the body orientation task, the results on a dataset named Human Orientation Classification (HOC) [Tosa] are reported. HOC is derived by the ETHZ [Sch, SD09] human re-acquisition dataset, representing pedestrians in different orientations and (background) conditions, captured by hand-held cameras. More precisely, the data was recorded using a pair of AVT Marlins mounted on a chariot, with a resolution of 640×480, a frame-rate of $13-14$ fps, and with a camera baseline of 0.4 meters. The images suffer from unbayering artefacts, slight motion blur, and sometimes missing contrast.

ETHZ is structured in three sequences of busy shopping streets for a total of 8555 images, each image 64×32 pixels containing a pedestrian. The images into 4 orientation classes (Front, Back, Left, and Right) are manually split, individuating a training and a testing partition. The dataset is complex due to the low resolution, the severe illumination artefacts, the occlusions and the consistent scale changes. According to the results in Fig. 4.2 regarding the content tensors, one could notice that EMI beats all the other tensor representations with an average accuracy of 68%. For what concerns the structural tensors, Fig. 4.3 shows that GRT beats $\mathrm{SST}_{\mathrm{struct}}$ with a 72%. This suggests that the set of tensor representation is more accurate with noisy data.

4.3.2 ViPER Human Dataset

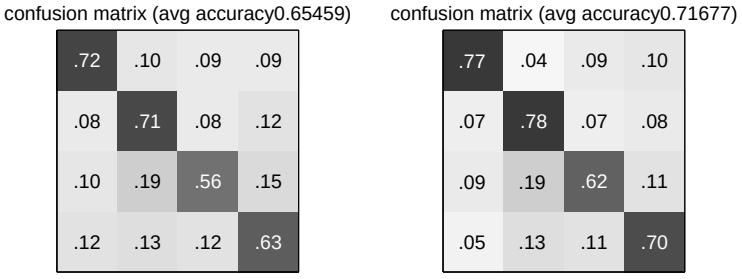
The ViPER human orientation dataset [Tosa] is derived from [GBT] and contains two camera views of 632 pedestrians. Each pair contains some images of the same pedestrian taken from different cameras, under different viewpoints, orientations and illumination conditions. All images are normalized to 128×48 pixels. Most



(a) (b)

(c)

Fig. 4.2. A comparison of content tensors on the HOC dataset. Confusion matrices showing the performances of COV (a), EMI (b), and $\mathrm{SST}_{\mathrm{content}}$ (c) tensor representations on the HOC dataset [Tosa].



(a) (b)

Fig. 4.3. A comparison of structural tensors on the HOC dataset. Confusion matrices showing the performances of $\mathrm{SST}_{\mathrm{struct}}$ (a) and GRT (b) tensor representations on the HOC dataset [Tosa].

of the examples contain a viewpoint change of 90 degrees. Since the task is the human orientation classification, the images of the two views are joined. Then, all the images are reflected vertically and small translations are performed to build a dataset of 8969 pedestrian images, finally. To build a balanced training set, about 1500 images are randomly sampled for each class and the testing set is composed of the remaining images. As in the previous case, the images into 4 orientation classes (Front, Back, Left, and Right) are manually split, individuating a training and a testing partition.

In this dataset the head appearance variability is very high, so, according to [EG10], it is difficult to build a reliable model, able to discriminate the front/back classes. It is necessary to remark that this fact does not affect the goodness of the other results in this Section, because all of the datasets are still challenging for other different reasons (light conditions, occlusions, etc.). Not surprisingly, the

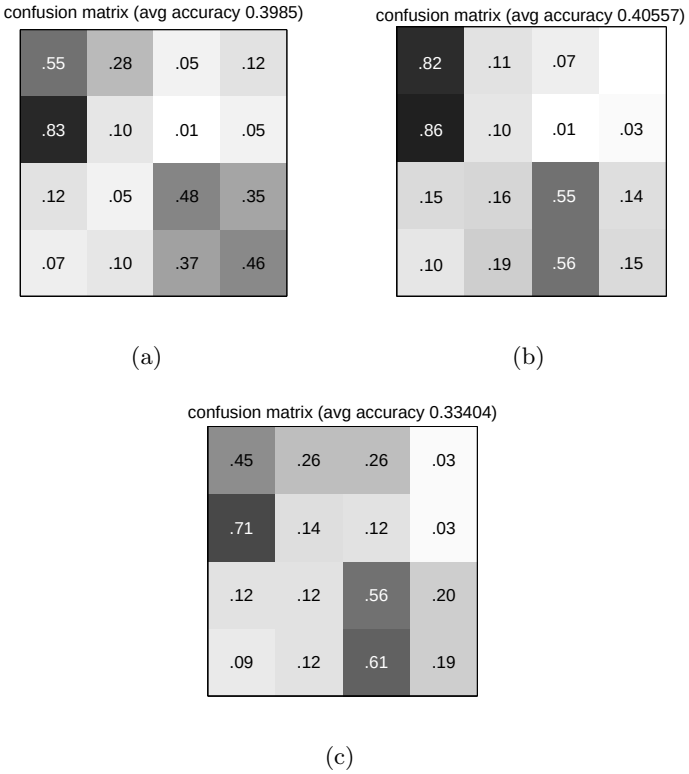
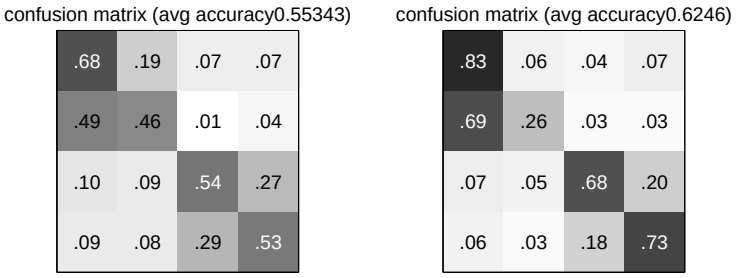


Fig. 4.4. A comparison of content tensors on the ViPER dataset. A Comparison of content tensors on the ViPER dataset. Confusion matrices showing the performances of COV (a), EMI (b), and $\mathrm{SST}_{\mathrm{content}}$ (c) tensor representations on the ViPER human orientation dataset [Tosa].

results in Fig. 4.4 have a low average accuracy with all the tensors; however it is interesting to highlight that, among the content tensors, EMI is the best with



(a) (b)

Fig. 4.5. A comparison of structural tensors on the ViPER dataset. Confusion matrices showing the performances of $\mathrm{SST}_{\mathrm{struct}}$ (a) and GRT (b) tensor representations on the ViPER human orientation dataset [Tosa].

41%, as in the previous case. Observing Fig. 4.5, one can notice that GTR beats $\mathrm{SST}_{\mathrm{content}}$ again, with 62% of average accuracy.

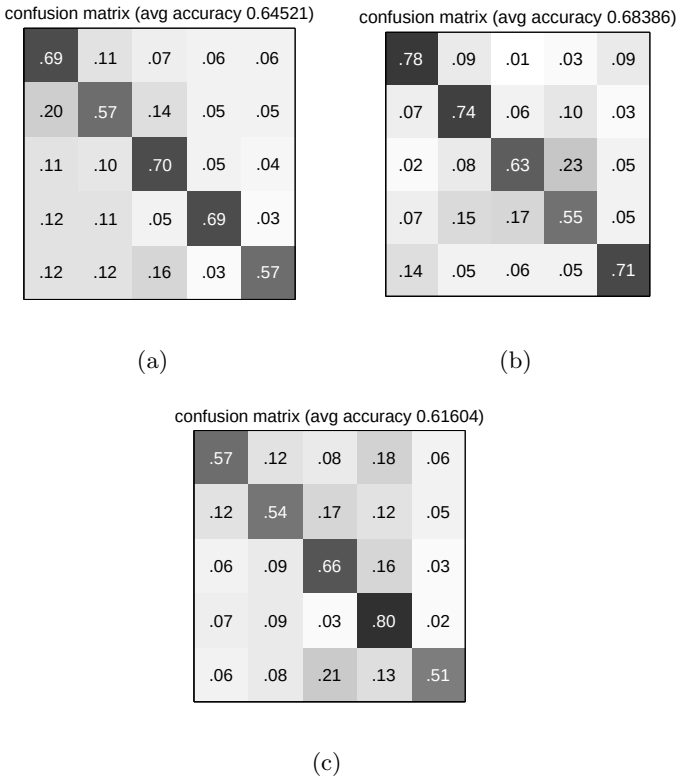
4.3.3 QMUL Head Dataset

The QMUL head dataset is formed by the head images taken from the i-LIDS dataset [Off08], which portrays an airport indoor scenario. To be more precise, i-LIDS consists of extensive CCTV footages of a busy underground scene captured under challenging lighting and viewing conditions. The video data are from two underground stations with video frame size of 640×480 recorded at 25 fps. Typically, the head image size varies from 60×60 to 10×10 pixels depending on the distance to the camera. They had been normalised to a size of 50×50. These scenes were crowded most of the time with many people, present at any given time. People were often under some degrees of occlusion and exhibited large head pose variations. Appearance variation of people includes beard faces, bold heads, light and dark hair and skin colours, all of which challenge modelling head/face image appearance with any assumption on clear-cut hair, skin and background segmentation.

The classification dataset obtained is composed of 18667 images, uniformly partitioned into 5 classes: Back (BA), Front (FR), Left (LE), Right (RI), and Background (BG). Background images contain portions of the background scene. The images are 50×50 pixels. The challenges of this dataset consist in scarce/non-homogeneous illumination, and quite severe occlusions. As for the HOC dataset (see Sec. 4.3.1), the results in Fig. 4.6 show that EMI achieves the best performances and that GRT outperforms $\mathrm{SST}_{\mathrm{struct}}$ (Fig. 4.7).

4.3.4 HIIT Head Dataset

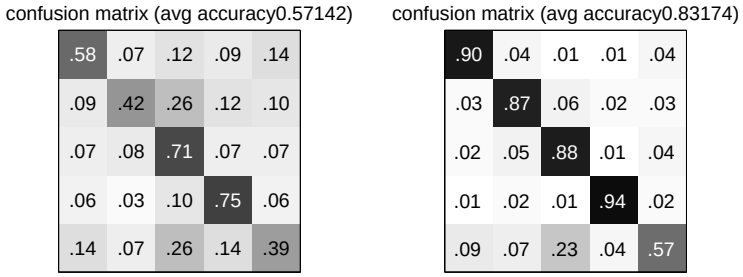
The HIIT head dataset has been built combining some indoor image data captured in a controlled scenario (a vision lab) and the Pointing04 [Gou], Multi-



(a) (b)

(c)

Fig. 4.6. A comparison between content tensors on the QMUL dataset. Confusion matrices showing the performances of COV (a), EMI (b), and $\mathrm{SST}_{\mathrm{content}}$ (c) tensor representations on the QMUL head dataset [TFC$^+$10].



(a) (b)

Fig. 4.7. A comparison between structural tensors on the QMUL dataset. Confusion matrices showing the performances of $\mathrm{SST}_{\mathrm{struct}}$ (a) and GRT (b) tensor representations on the QMUL head dataset [TFC$^+$10].

PIE [GMC+07], and QMUL [Tosa] datasets. The Pointing04 and the Multi-PIE [GMC+07] are briefly described to characterise completely the HIIT data.

The Pointing04 corpus was included as part of the Pointing 2004 Workshop on Visual Observation of Deictic Gestures to allow the uniform evaluation of head pose estimation systems. It was also used as one of two datasets to evaluate head pose estimators in the 2006 International Workshop on Classification of Events Activities and Relationships (CLEAR'06) [SBB+06]. Pointing04 consists of 15 sets of near-field images, with each set containing 2 series of 93 images of the same person at 93 discrete poses. The discrete poses span both pitch and yaw, ranging from $-90°$ to $90°$ in both DOF. The subjects range in age from 20 to 40 years old, five possessing facial hair and seven wearing glasses. Each subject was photographed against a uniform background, and the heads were manually cropped. Head pose ground truth was obtained by directional suggestion. The CMU Multi-PIE (Pose, Illumination, and Expression) face database contains more than 750000 images of 337 people recorded in up to four sessions over the span of five months. Subjects were imaged under 15 view points and 19 illumination conditions while displaying a range of facial expressions.

The resulting (HIIT) dataset has 6 classes, each composed of 2000 examples. The size of the examples is 50×50 pixels, without any margins around the heads. The main characteristic of this dataset is that it has a stable background and no occlusions. However, in this dataset the facial appearance of the people varies significantly in a number of factors, including identity, illumination, pose, and expression. Therefore it represents the ideal scenario where to evaluate how an head orientation classifier is robust to changes in the appearance. The results reported in Fig.4.8 show the power of the COV tensor in ideal conditions, where the examples in the dataset are not noisy, even if the appearance variability is high. In fact it outperforms with 79% of average accuracy all the other representations. On the other hand, for what concerns the structural tensors, Fig.4.9 shows the power of the GRT tensor in its natural applicative scenario. In fact, in this case GRT tensor leads to an average accuracy of 93%.

4.3.5 CIFAR10 Object Dataset

The CIFAR10 is a dataset used for object classification and it has 10 object categories, namely aeroplane, bird, car, cat, deer, dog, frog, horse, ship, and truck. The training set has 5000 examples per class, while the test set has 1000 examples per class. The 32×32 resolution of the images in the dataset and their variability make recognition very difficult. In fact a traditional method, based on features extracted at interest points, does not work. Observing the results in Fig. 4.10, even considering the challenging object recognition task, EMI confirms its superior performances. On the other hand, $\mathrm{SST}_{\mathrm{content}}$ performs poorly. However, the performance of $\mathrm{SST}_{\mathrm{content}}$ is extremely different if the set of image features used to build the tensor has been changed. In particular, augmenting the number of features adopting a feature set learned by a Restricted Boltzmann Machine (RBM) [BdFL+11], $\mathrm{SST}_{\mathrm{content}}$, one can obtain the best average classification accuracy, if compared to EMI and COV tensors. Fig. 4.12(a)(b)(c) shows the results using the feature set obtained convolving the filters depicted in Fig. 4.12(d) with the dataset images to build the tensor representations.

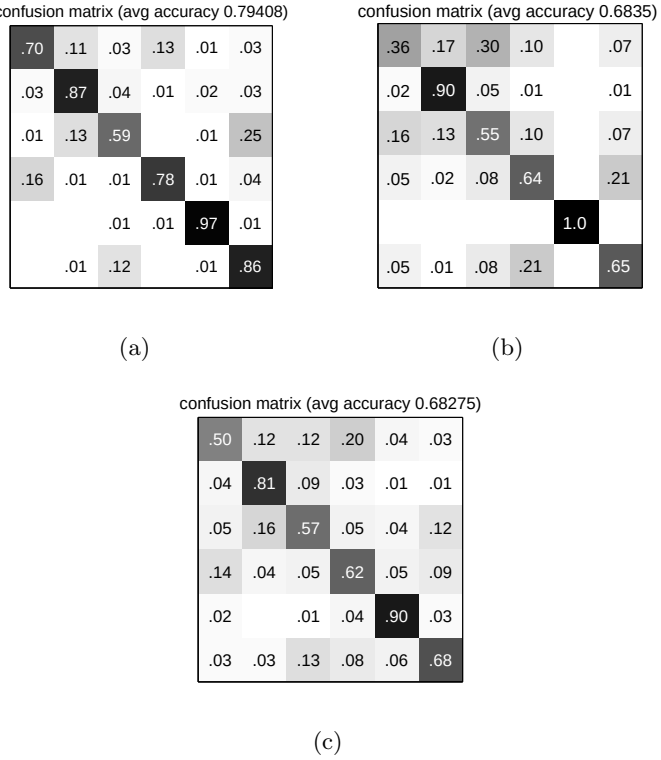


Fig. 4.8. A comparison of content tensors on the HIIT dataset. Confusion matrices showing the performances of COV (a), EMI (b), and $\mathrm{SST}_{\mathrm{content}}$ (c) tensor representations on the HIIT head dataset [Tosa].

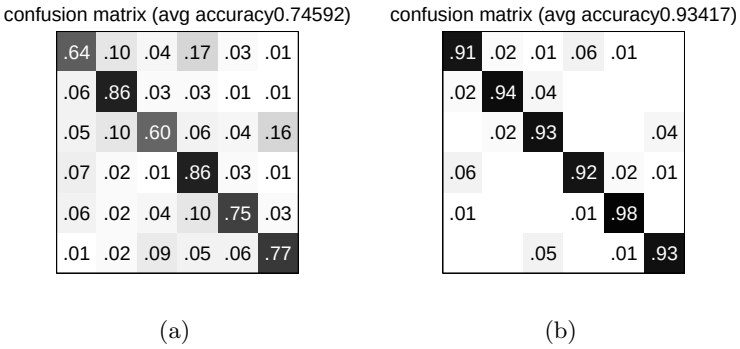
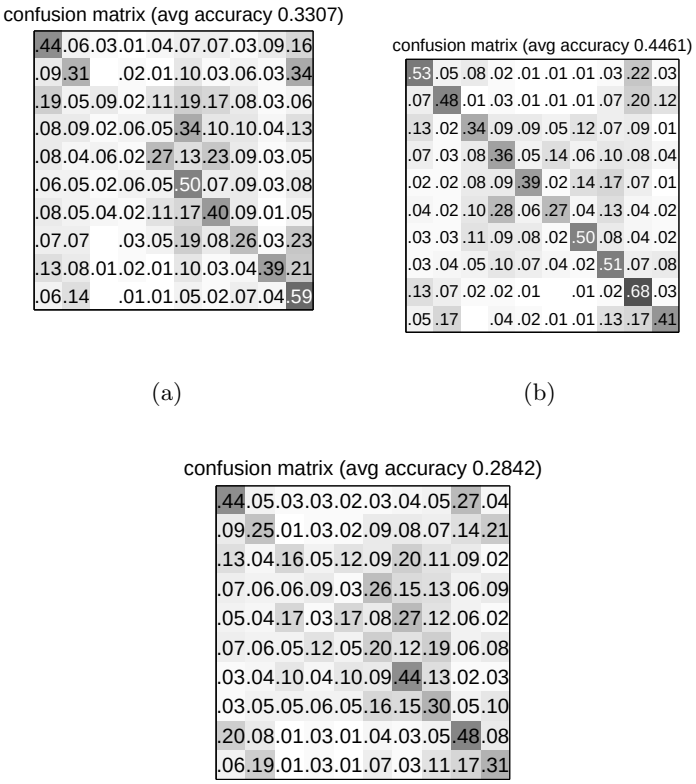


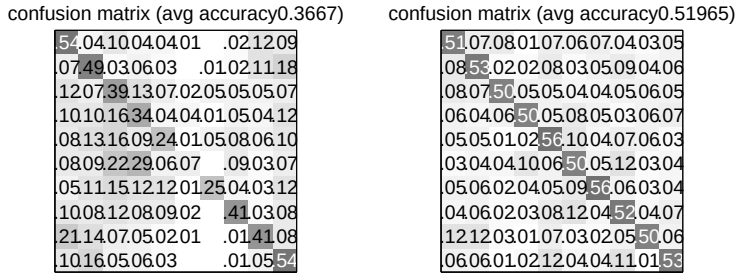
Fig. 4.9. A comparison of structural tensors on the HIIT dataset. Confusion matrices showing the performances of $\mathrm{SST}_{\mathrm{struct}}$ (a) and GRT (b) tensor representations on the HIIT head dataset [Tosa].



(a) (b)

(c)

Fig. 4.10. A comparison of content tensors on the CIFAR10 dataset. Confusion matrices showing the performances of COV (a), EMI (b), and $\mathrm{SST}_{\mathrm{content}}$ (c) tensor representations on the CIFAR10 object dataset [KH06].



(a) (b)

Fig. 4.11. A comparison of structural tensors on the CIFAR10 dataset. Confusion matrices showing the performances of $\mathrm{SST}_{\mathrm{struct}}$ (a) and GRT (b) tensor representations on the CIFAR10 object dataset [KH06].

For what concerns structural tensors, GRT with 52% show the best performance as for all of the previous cases (see Fig. 4.11). However, the computational cost of GTR is much higher than $\mathrm{SST}_{\mathrm{struct}}$.

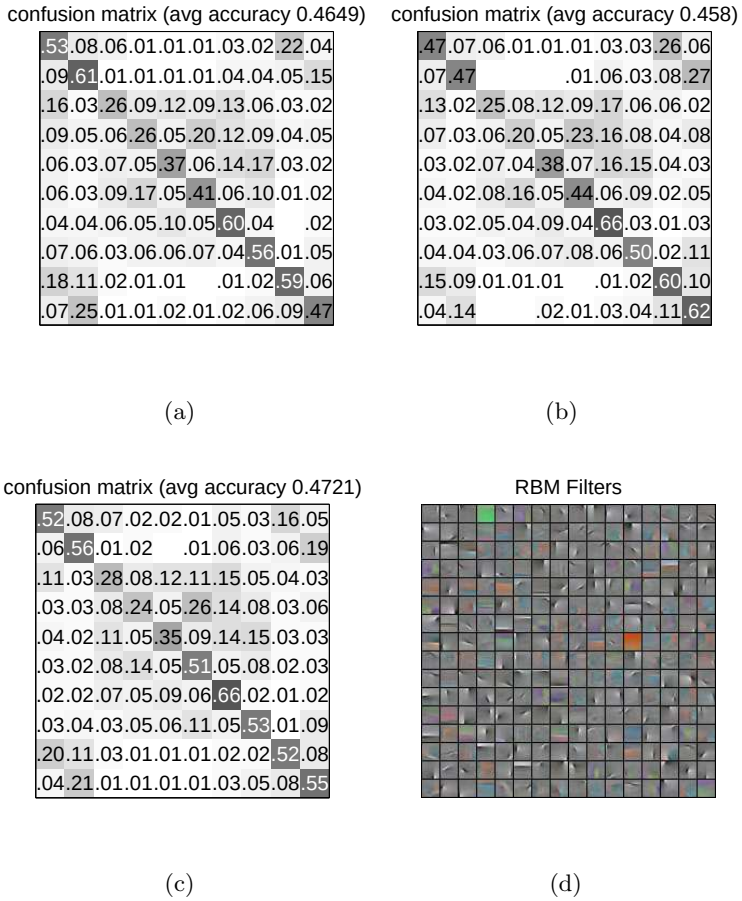


Fig. 4.12. A Comparison between content tensors on the CIFAR10 dataset using RBM features. Confusion matrices showing the performances of COV (a), EMI (b), and $\mathrm{SST}_{\mathrm{content}}$ (c) tensor representations using RMB features (d) [BdFL$^+$11] on the CIFAR10 object dataset [KH06].

5

Detection using Tensors

Contents

5.1 Introduction

This Chapter is focused on the object detection task and in particular the pedestrian detection task. This because people are among the most important "objects" that can be detected from images and videos. Detecting and tracking people are thus important areas of research, and computer vision is bound to play a key role.

Applications include robotics, entertainment, surveillance, and care for the elderly and disabled.

To represent a person COV (covariance) tensors (introduced in Sec. 4.2.1) and SST tensors (i.e. $\mathrm{SST}_{\mathrm{struct}}$) (described in Sec. 4.2.3) are utilized. The main contributions of this chapter are the following: four object architectures and the relatives learning frameworks are outlined; i.e. a framework based on automatic feature selection made using Boosting which improves the state-of-the-art pedestrian detector [TPM08], a part-based pedestrian detector on multiple tangent spaces (one for every part) based on COV tensors, a low resolution pedestrian detector based on $\mathrm{SST}_{\mathrm{struct}}$, and a robust to occlusion set-based pedestrian detection framework, where the body configuration is not fixed.

In particular, in Sec. 5.2 a fast machine learning framework derived from [TPM08] is described: it is able to manage the covariance matrices as OI (Objects of Interest) descriptors into a binary boosting classification framework. This work shows that the detection performances of the state-of-the-art approach [TPM08], which combines boosting and the use of covariance matrices, can be greatly improved, from both a computational and a qualitative point of view, by considering practical and theoretical issues, and allowing also the estimation of occlusions in a fine way.

The previous work introduces different contributions that are useful to speed up the model training phase and the accuracy of the final detection. However, since the covariances are embedded in a boosting framework which selects them in an unsupervised way, the model remains quite expensive to be computed. In fact, thousands of covariances are selected to build the final detector. Since the OIs are the people (i.e. pedestrians), it is possible to exploit the human knowledge to impose a fixed human part-based layout which can be composed by few semantic parts. In particular, to model a human a hierarchy of fixed overlapped parts is adopted; each part is described by COV tensors. Each part is modelled by a boosted classifier, trained using boosting on different tangent spaces of a Riemannian manifold (see. 2.4.3 for details). All the classifiers are then linked to form a high-level classifier, through weighted summation, whose weights are estimated during the training phase. This classification approach, described in Sec. 5.3, is simple, light and robust.

In Sec. 5.4 $\mathrm{SST}_{\mathrm{struct}}$ tensors can be used to measure the self-similarity of the parts of a human body for the detection task. So, a framework for this task is proposed, where pedestrians can be at very low resolution. Since parts are tricky to be modelled from low resolution images, a pyramidal regular grid of patches [LSP06] is adopted. Then it is shown how $\mathrm{SST}_{\mathrm{struct}}$ beats the COV tensor representation for the low resolution pedestrian detection task on a state-of-the-art pedestrian detection benchmark.

In Sec. 5.5 the attention is focused on the capability to detect people in crowded scenes. In fact, if people detection is performed in a non-problematic scenario (e.g. one where people are not occluded, with a limited range of scales and pose variations), a lot of effective existing frameworks can be used to solve this task. On the other hand, if the scenario is problematic, only few of these systems are really useful. Since none of the above-mentioned frameworks is able to solve all of the previous problems, a unified framework capable to jointly cope with those

issues is studied. Therefore, the goal is to detect as many people as possible even when the human body layout cannot be inferred.

In Sec. 5.6 an effective, low-latency, affordable detection architecture is proposed, especially suited for embedded platforms. In particular, a highly-parallelizable classification framework is designed for an implementation based on FPGA (Field Programmable Gate Array) implementation, which is suitable for generic detection problems. The underlying model consists in a weighted sum of boosted binary classifiers, trained on a set of overlapped image patches. Each patch is described by estimating the COV tensor of a set of image features. COV tensors live on Riemannian Manifolds (see Sec. 2.6 for details), and can be approximated in the Euclidean Vector Space in a cheap and conservative way. The hardware design has been developed in parallel and with specific architectural solutions, allowing a fast response without degrading the functional performances. The proposed architecture has been finally specialized in the challenging pedestrian detection problem.

Sec. 5.7 shows the results of the human detection module developed for the SAMURAI (Suspicious and Abnormal behaviour Monitoring Using a netwoRk of cAmeras) project, in which some of the previous introduced frameworks are implemented. The project aims to develop and integrate an intelligent surveillance system for robust monitoring of both inside and surrounding areas of a critical public infrastructure. SAMURAI employs networked heterogeneous sensors, so that multiple complementary sources of information can be fused to create a visualisation of a more complete "big picture" of a crowded public space.

5.2 Fast Unsupervised Covariance Tensor Selection for Pedestrian Detection

In this Section, draw attention to the state-of-the-art method in [TPM08], where a human is modelled by covariance matrices of image features. This representation is convenient: covariances allow a great robustness, regarding for example the number of elements employed for its calculus, i.e. the size of the pedestrian. The method reaches its best performances on the INRIA dataset [Dal05], but such tools have a considerable computational burden. In fact, covariance matrices, which belong to Sym^+ 2.6.1, live in a Riemannian manifold 2.5. This implies a high effort to compute all the operators needed in the boosting framework.

A set of improvements is proposed, that tackle the framework of [TPM08] under both a theoretical point of view, managing the Riemannian geometry in a finer and economic way, and a practical point of view, suggesting tricks that lead to a more robust and faster detection framework, also able to finely model occluded individuals.

The proposed improvements are: extracting candidate weak classifiers using an a priori probability distribution on the human shape (Sec. 5.2.2.1); building the training set based on a *low level semantic* that decreases the cascade [VJ01] complexity (Sec. 5.2.2.2); working more efficiently on the space of the covariance matrices, using hybrid operators (Sec. 5.2.2.3); creating a more effective weak classification method, based on polynomial regression (Sec. 5.2.2.4).

5.2.1 Binary Classification on Riemannian Manifolds

First the binary LogitBoost on Riemannian manifold [TPM08], that extends the standard LogitBoost (see Alg. 2), is briefly introduced.

Let $\{\mathbf{X}_i, y_i\}_{i=1,\dots,N}$ be the set of training examples of fixed size, with labels $y_i \in \{1 = \text{human}, 0 = \text{other}\}$ and $\mathbf{X}_i \in \mathcal{M}$, i.e. the Riemannian manifold of covariance matrices. The goal is to find a strong classifier $F(\mathbf{X}_i) : \mathcal{M} \mapsto \{0, 1\}$, formed by an ensemble of weak learners (WLs), that partitions the input space into 2 classes, according to the labelling. The probability of $\mathbf{X}_i$ being in class 1 is represented by

$$p(\mathbf{X}_i) = \frac{e^{F(\mathbf{X}_i)}}{e^{F(\mathbf{X}_i)} + e^{-F(\mathbf{X}_i)}}, \tag{5.1}$$

where

$$F(\mathbf{X}_i) = \sum_{l=1}^{N_l} f_l(\mathbf{X}_i), \tag{5.2}$$

and $\{f_l\}_{l=1,\dots,N_l}$ is the set of WLs. Iteratively, each WL is selected by fitting a weighted least-square regression function f_l of training points $\mathbf{X}_i$ to response values z_i and weights w_i:

$$f_l = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{N} w_i |z_i - f(\mathbf{X}_i)|^2. \tag{5.3}$$

where $\mathcal{F}$ is the set of possible WLs. Note that each WL f_l focuses on a patch b_l, whose size and position over all data are selected by evaluating a bunch of candidate sizes and positions, sampled uniformly over the pedestrian image.

5.2.2 Improvements

5.2.2.1 Towards a faster, more informative WLs selection

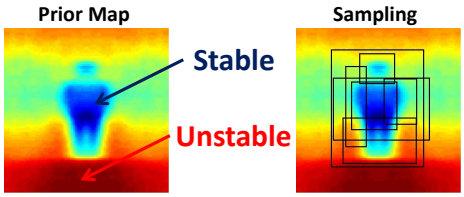


Fig. 5.1. OPT 1. A prior map (on the left) is built on which stable regions are highlighted. WLs are selected (on the right) sampling this prior distribution over the whole pedestrian image.

The process of selection of the patches b_l is accelerated, by sampling only over *interesting* position values, i.e. those pixels representing people with higher probability. This means exploiting a prior map of human appearance. A subset of positive samples is built, masking with 1 the pedestrian and 0 the background, and computing the per-pixel mean. Normalizing the result by the number of positives the prior map is provided. The addition goes beyond the mere acceleration. In fact, it minimizes the selection of patches on the background area, that can be discriminative in an erroneous way. For example, if the positive dataset depicts people with a similar background, whose visual layout differs from the content of the negative dataset, the background information is very discriminative and it has to be selected by the WLs. This makes a classifier incapable of generalizing about different backgrounds. This optimization is referred as OPT 1 and it is depicted in Fig. 5.1.

5.2.2.2 Avoiding the overtraining

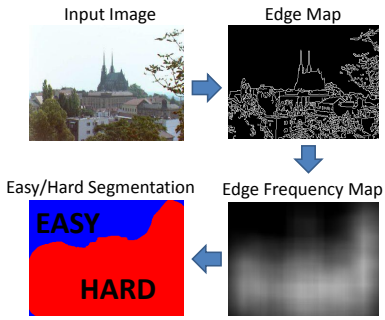


Fig. 5.2. OPT 2. To avoid the overtraining an ordered training set of negative examples is built according to a easy/hard negatives segmentation.

Building the negative set $\{B_i\}_{i=1,\ldots,N_{BG}}$ is very compelling, being it representative of everything but humans. The B_i negative samples are fed gradually and randomly into the training of the classifier, exploiting the classic boosting cascade structure [VJ01] that allows a very fast classification. Considering these facts a strategy to use the negative samples is devised. They are ordered by *difficulty*: *easy* negatives are clearly different to humans, while *hard* negatives are not. Experimentally, I discovered that the negative examples harder to classify are characterised by a high textural or structural content. Therefore, a criterion regarding difficulties can be based on the high frequency content of the images is proposed. For each B_i, a map containing the edge response is built to compute the number c_i of pixels whose, edge response is above a threshold τ. Sorting the B_is according to c_i allows to assign a difficulty score to the samples. During the learning phase, one can adopt this ordering to feed the negative examples to the classifier (OPT 2) and it is depicted in Fig. 5.2. This permits first to construct simple decision boundaries, and then to build the more complex ones. This strategy

decreases the risk of overtraining and improves the efficiency of both the learning and the detection phase, because the simplest negatives are filtered out very quickly.

5.2.2.3 More efficient analysis on $\mathcal{M}$

In the Riemannian manifold $\mathcal{M}$, there are three fundamental operations needed for boosting purposes. The first one is a mapping $\log_{\boldsymbol{\mu}_l} : \mathcal{M} \mapsto \mathbb{R}^n$, called logarithmic map, that projects the input covariance matrices into the vector tangent space at a point $\boldsymbol{\mu}_l \in \mathcal{M}$; in the tangent space, standard WLs like regressors or linear discriminants can be estimated. The second operation is the inverse mapping $\exp_{\boldsymbol{\mu}_l} : \mathbb{R}^n \mapsto \mathcal{M}$, called exponential mapping. The third one is the centroid calculation, i.e. the operator selecting the projection point $\boldsymbol{\mu}_l$, that is the mean of an arbitrary set of points on $\mathcal{M}$.

The affine-invariant Riemannian framework of [PFA06] deals with Sym^+ matrices. On one hand, the log and exp mapping can be easily computed exploiting the Sym^+ matrices properties. On the other hand, the calculation of centroid has no closed form, that makes it very slow.

Recently, a novel metric family called Log-Euclidean is proposed in [AFPA08]; they are similarity-invariant and have a closed form for the computation of the centroid $\boldsymbol{\mu}_l$. However, the computation of $\log_{\boldsymbol{\mu}_l}$ and $\exp_{\boldsymbol{\mu}_l}$ with these metrics is tricky and expensive, involving matrix differential calculations.

Before to present the improvement, in Tab. 5.2.2.3 the influence of the different metrics on the basic operation on a Riemannian manifold is summarized. Let $\mathbf{X}_1, \mathbf{X}_2 \in Sym_d^+$, $\mathbf{x} \in Sym_d$ the tangent vector in $T_{\mathbf{X}_1} Sym_d^+$ associate with the unique geodesic between $\mathbf{X}_1$ and $\mathbf{X}_2$. The log and exp maps have a well known

Euclidean	Log-Euclidean [AFPA08]	Natural 2.6.1
$\mathbf{x} = \mathbf{X}_1 - \mathbf{X}_2$	$\mathbf{x} = \log_{\mathbf{X}_1}(\mathbf{X}_2)$	$\mathbf{x} = \log_{\mathbf{X}_1}(\mathbf{X}_2)$
$\mathbf{X}_2 = \mathbf{X}_1 + \mathbf{x}$	$\mathbf{X}_2 = \exp_{\mathbf{X}_1}(\mathbf{x})$	$\mathbf{X}_2 = \exp_{\mathbf{X}_1}(\mathbf{x})$
$\mathrm{dist}(\mathbf{X}_1, \mathbf{X}_2) = \|\|\mathbf{X}_1 - \mathbf{X}_2\|\|$	$\mathrm{dist}(\mathbf{X}_1, \mathbf{X}_2) = \|\|\mathbf{x}\|\|_{\mathbf{X}_1}$	$\mathrm{dist}(\mathbf{X}_1, \mathbf{X}_2) = \|\|\mathbf{x}\|\|_{\mathbf{X}_1}$
$\frac{1}{N}\sum_i \mathbf{X}_i$	$\exp_{\mathbf{I}_d}(\frac{1}{N}\sum_i \log_{\mathbf{I}_d}(\mathbf{X}_i))$	$\exp_{\mu^t}(\frac{1}{N}\sum_i \log_{\mu^t}(\mathbf{X}_i))$
swelling effect	similarity-invariant	affine-invariant

Table 5.1. Basic operations on a Riemannian Manifold.

formulation for Natural metric, while are complicated for Log-Euclidean metric. Moreover, from the table it easy to understand why the Euclidean is not considered. In fact it leads to the *swelling* effect: the determinant of the Euclidean mean of tensors can be larger than the determinants of the original tensors, which is physically unrealistic.

Consequently, the idea is to combine the similarity-invariant and affine-invariant frameworks (Opt 3) to gain efficiency. The log and exp operators are computed in the affine-invariant way as in [PFA06]. $\boldsymbol{\mu}_l$, instead, is calculated in the similarity-invariant way as:

$$\boldsymbol{\mu}_l = \exp\left(\frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i \log(\mathbf{X}_i)\right), \tag{5.4}$$

where $\log(\mathbf{X}) = \mathbf{U}\log(\mathbf{D})\mathbf{U}^T$ with $\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{U}^T$ is the eigenvalue decomposition of $\mathbf{X}$, and $\log(\mathbf{D})$ is the diagonal matrix composed of the eigenvalues' logarithms. The following equivalences are highlighted:

$$\log(\mathbf{X}) = \log_{\mathbf{I}_d}(\mathbf{X}), \quad \mathbf{X} \in Sym^+ \tag{5.5}$$

$$\exp(\mathbf{x}) = \exp_{\mathbf{I}_d}(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^n, \tag{5.6}$$

where $\mathbf{I}_d \in Sym^+$ is the identity matrix. This fact comes from the corollary 3.7 in [AFPA08], where the equivalence of the tangent space at the identity matrix of Sym^+ and Sym is proved. This means that computing $\boldsymbol{\mu}_l$ as in Eq. (5.4) implies to work on the Euclidean space of symmetric matrices Sym. Points from Sym^+ can be mapped to Sym simply using the log operator. The result obtained on Sym is then mapped back to the Sym^+ domain with the exponential map. Thanks to this formulation, a centroid can be calculated approximately 20 times faster than using the formulation in [PFA06].

5.2.2.4 More powerful WLs

The type of WLs that form the boosting ensemble is carefully analysed. In [TPM08], the authors employ linear regression functions, suggesting that a further study on this aspect would be useful. In this analysis, after a preliminary study on several WLs, the polynomial functions are selected (Opt 4). In fact, some WLs, as for example linear regression functions, are unable to represent complex decision boundaries, while complex WLs, as for example piecewise constant regression functions, quickly lead to overfitting. This class of WLs has several advantages: it is easily implementable, efficiently computable and flexible. In fact, weighted multidimensional polynomial fitting can be formalized as a linear problem [MSD97]. A k-th degree polynomial in $\mathbb{R}$ is defined as follows:

$$y = a_0 + a_1 x + \ldots + a_k x^k, \tag{5.7}$$

where $y, a_0, \ldots a_k, x, \ldots x^k \in \mathbb{R}$. The matrix form for the least-square fit can be obtained by writing the Vandermonde matrix as a linear system:

$$\begin{bmatrix} 1 & x_1 & \cdots & x_1^k \\ \vdots & x_2 & \ddots & x_2^k \\ 1 & x_n & \cdots & x_n^k \end{bmatrix} \begin{bmatrix} a_0 \\ \vdots \\ a_k \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_k \end{bmatrix}. \tag{5.8}$$

Eq. (5.8) in matrix notation is

$$X\,\mathbf{a} = \mathbf{y}, \tag{5.9}$$

where each row in X represents an example of the training set. Generalizing, the matrix form to a k-th order polynomial in $\mathbb{R}^n$, with no mixed terms. The least square fit could be formalised as a linear system:

$$\begin{bmatrix} X_1 \ldots X_N \end{bmatrix} \begin{bmatrix} \mathbf{a}_1 \\ \vdots \\ \mathbf{a}_N \end{bmatrix} = \begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_N \end{bmatrix}. \tag{5.10}$$

After different trials, in the experiments the second degree polynomials have always been considered.

5.2.2.5 Occlusion modelling by WLs analysis

This feature (Opt 5) aims to refine a positive detection output, highlighting when and where the detected person is occluded by an object (the modelling of occlusions caused by people is already faced in [TPM08]). This helps whenever the mere detection is followed by further analysis. In people re-identification, for example, the availability of genuine person's details, minimizing the clutter, is very useful. The idea is to analyse the responses of the WLs, looking for possible agglomerations.

In detail, the presence of 4 different synthetic occlusions is tested (see Fig. 5.3(a)): TOP, BOTTOM, LEFT, RIGHT, parametrised by the size value s. The process

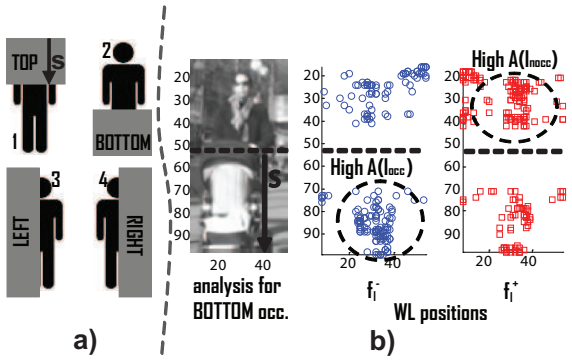


Fig. 5.3. On the left, the occlusions used; on the right, WL responses for the image in the centre.

exploits the fact that each WL f_l focuses on a patch b_l, that is, a f_l judges a *portion* of the test image. Therefore, a compact, localized cluster of WLs, whose responses are positive, will indicate a human part, with high probability. On the contrary, a set of WLs with negative responses will probably indicate an occluding object. As shown in Fig.5.3(b), each value of s determines a bipartition of the images, I_{occ} and I_{nocc}, where I_{occ} (I_{nocc}) is the occluded (not occluded) part. On I_{occ}, the *partial agreement*

$$A(I_{occ}) = ||f_l^-||/||f_l^+ + f_l^-||, \tag{5.11}$$

that is the percentage of WLs in I_{occ}, is computed, not overlapping with the border instantiated by s (the dotted line in Fig. 5.3(b), whose response is negative. A similar reasoning holds for $A(I_{nocc})$. The two measures are combined

$$BS = A(I_{occ}) + A(I_{nocc}). \tag{5.12}$$

Maximising BS over s for a single kind of occlusion gives s_{best}, i.e. the best occlusion size. The comparison of s_{best}s of each kind of occlusion gives the most

probable occlusion. Experimentally, a threshold BS_τ is fixed and below that BS does not represent an occlusion. This is reasonable, since the distribution of the WLs positive (negative) responses is uniform for a non occluded object.

5.2.3 Experimental Results

The proposed human detector is trained on INRIA Person dataset, that contains 1774 images portraying humans, doubled through mirroring, and 1671 person-free images, all of size 64×128. The setting for the training phase is the same of [TPM08]. Both the proposed approach and the original [TPM08] are implemented with Matlab on an Intel 2.83 Ghz processor with 4 Gbytes of RAM. The training takes three days on average with the proposed approach and more than two weeks with the original approach.

First, the effects of each proposed improvement with respect to [TPM08] is shown, on a randomly chosen subset of the INRIA dataset (500 positive and 1000 negative examples) in a cascade of 10 levels. The performance by computing the Detection Error Tradeoff (DET) curve is measured. It shows the tradeoff between true and false positives on a log-log scale. The results are on Fig. 5.4 (top). The y-axis corresponds to the miss rate

$$\text{FalseNeg}/(\text{FalseNeg} + \text{TruePos})$$

, and the x-axis corresponds to the false positives

$$\text{FalsePos}/(\text{FalsePos} + \text{TrueNeg})$$

, in this case the False Positives Per tested Window (FPPW). All the improvements Opt 1, 2, 3, and 4 provide an improvement in accuracy with respect to [TPM08].

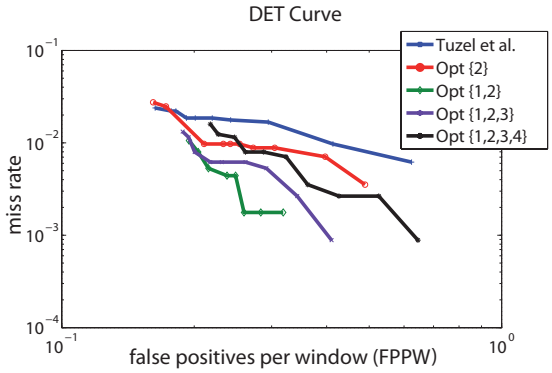


Fig. 5.4. Comparison on the restricted dataset, adding one-by-one the Opts.

Afterwards, a rejection cascade of 30 levels is trained using the whole INRIA dataset, reproducing the system of [TPM08]. The effects of the proposed policy in selecting the examples for the cascade are evident in Fig. 5.5, that shows the number of WLs per level. The proposed improvements produce a cut of the cascade

complexity, evaluated as number of classifiers, of the 15% and 58%, using the linear and the polynomial regression, respectively. This results in a faster learning and testing phase.

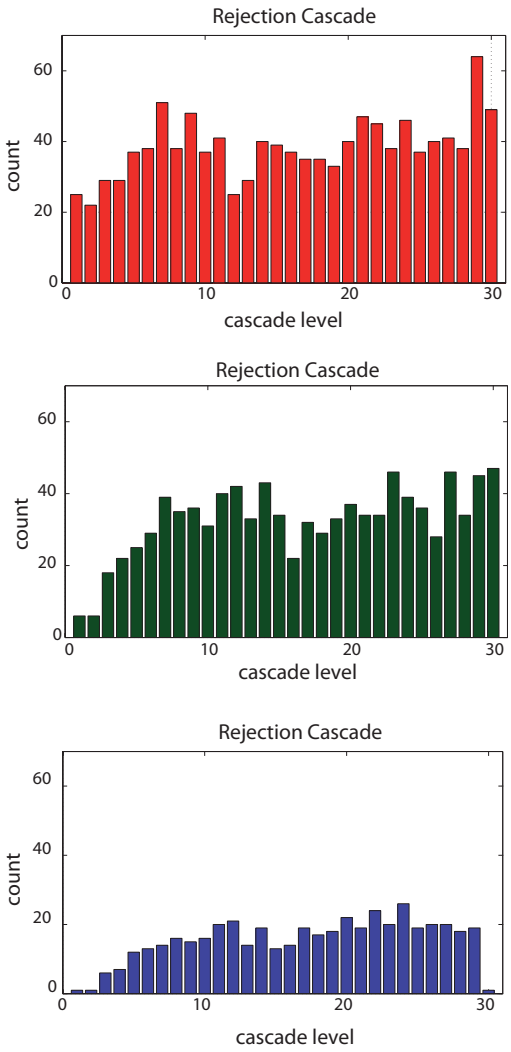


Fig. 5.5. Comparison between cascades of 30 levels on the INRIA dataset. Top: random selection of negative samples, as in [TPM08]. Centre: OPT 1,2,3 are applied. Bottom: all optimizations are exploited.

In Fig. 5.6 the framework is compared to the state-of-the-art in terms of DET curve. Both the proposed detectors, using linear and polynomial regressors, have good generalization abilities, in a slightly different way. Indeed, one may notice that in the linear case at all cascade levels, indicated by the markers, better per-

formances in terms of miss rate are achieved, while the same performance as the original approach in terms of false positives is maintained. The polynomial case is instead close to the original approach in terms of miss rate, but it is the fastest approach respect to the others.

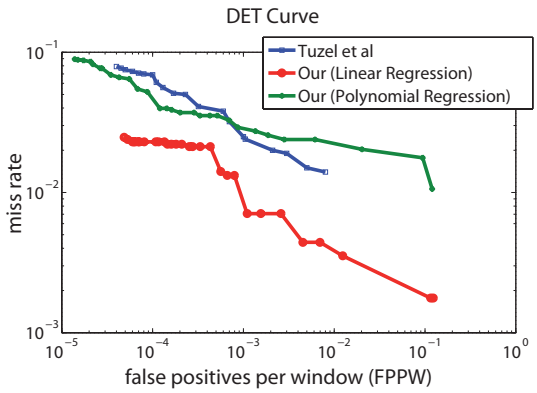


Fig. 5.6. Comparison between [TPM08] and the proposed method (with and without Opt 4), on the complete dataset.

Concerning the occlusion modelling (Opt 5), the occlusions of 200 positive detections are evaluated qualitatively, validating their correctness by subjective judgement (no ground-truth data is available). An accuracy of 81% is reached, where each occlusion detection can be correct (= 1) or not (= 0). Some results are shown in Fig.5.7.

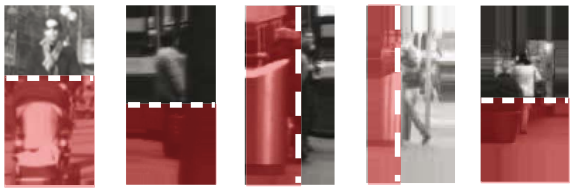


Fig. 5.7. Five examples of occlusion modelling: in red the parts **detected** as occlusions.

5.3 Part-based Pedestrian Detection on Multiple Tangent Spaces

Robust object detection is important for many applications. In particular, in the context of video surveillance, pedestrians are a very important and very challenging class of objects to detect. Among the recent approaches proposed in literature,

part-based models [MSZ04, WN09, YTCC09] seem to provide the best performances, as shown in [DWSP09]. This is because these models are intrinsically robust to partial, inter-object occlusions.

Following the same direction, a new part-based model for pedestrian detection is proposed. The parts are hierarchically structured, and a priori fixed, as in [WN09]. Each part is described by a COV tensor, that encodes information of the variances of a set of defined features inside a region (patch), along with their correlations and the spatial layout. This descriptor is robust to illumination and scale variations.

In this Section, I claim that i) injecting a priori knowledge about the human structure by suggesting the parts to be focused and ii) thanks to an adequate training of such parts by boosting via polynomial fitting, the feature selection phase is not necessary. The resulting framework is light (the computational cost of the training phase is dramatically reduced with respect to [TPM08]), and it outperforms the state-of-the-art methods on the INRIA person dataset.

The rest of the Section is organized as follows. Sec. 5.3.1 presents the architecture of the proposed classification system on Riemannian manifolds. Practical details and experimental results are explicated in Sec. 5.3.2.

5.3.1 System architecture

Inspired by [WN09], the human body is divided into parts, according to their semantic meaning (head, torso, etc.). These parts are then organised in a hierarchy of three levels, for a total number of eleven parts (see Figure 5.8).

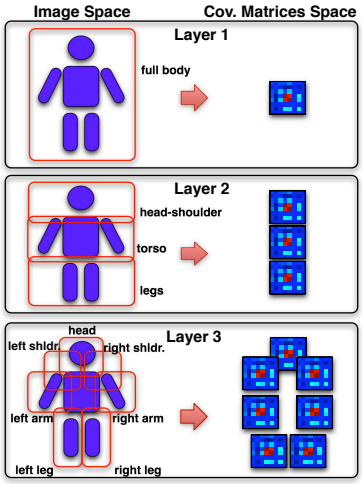


Fig. 5.8. Part-based human model. The human body is hierarchically divided into 11 parts, and each part is described by a covariance matrix descriptor.

A covariance descriptor is associated with each part, and it is estimated as follows. As in [TPM08], for each pixel (x, y) inside the region, a bunch of information

is gathered about that pixel into a feature vector:

$$[\ x\ y\ |I_x|\ |I_y|\ \sqrt{I_x^2 + I_y^2}\ |I_{xx}|\ |I_{yy}|\ \arctan\frac{I_x}{I_y}\ ]^T, \tag{5.13}$$

where I_x, I_{xx}, etc. are intensity derivatives and the last term is the edge orientation. From these vectors their covariance matrix can be estimated. This operation is done efficiently using integral images [TPM06].

Given the descriptors, the system is composed of two phases: first, each body part is trained separately, using LogitBoost; second, the part classifiers are combined together. These two phases are detailed in the following paragraphs.

5.3.1.1 Phase 1: boosting the part models

Let $\{\mathbf{X}_{ip}, y_i\}_{i=1,\ldots,N}$ be the set of training examples (COV tensors), of a fixed human part p. These examples are points in the Riemannian manifold $\mathcal{M}$. Training a classifier on $\mathcal{M}$ using a boosting approach implies tp project all points into the local tangent space $T_{\mathbf{X}}$ of a point $\mathbf{X} \in \mathcal{M}$. $T_{\mathbf{X}}\mathcal{M}$ is a Euclidean space (so that a standard classification algorithm can be employed on the projected points). In [TPM08], the authors empirically show that a good choice of $\mathbf{X}$ is the Karcher mean $\boldsymbol{\mu}_p$ of $\{\mathbf{X}_{ip}\}_{i=1,\ldots,N}$, i.e. the point that minimizes the sum of squared Riemannian distances.

The framework proposed in [TPM08] is a greedy algorithm, where at each boosting iteration the most discriminative patch inside the detection window, i.e. the patch on which a single weak classifier gives the best classification performance, is selected. This implies having several covariance descriptor sets, corresponding to each of the possible patches, projecting them into their tangent spaces $T_{\boldsymbol{\mu}_p}$ and choosing the one where positive and negative examples are better separated.

A different direction is followed, which is simpler, less computationally expensive, and gives good performances at the same time. The classifier can be instructed about which are the most interesting (discriminative) areas for the human body, thus concentrating on classification rather than feature selection. This means that a strong classifier is built for each of the body parts, so that the final human detector is the composition of a few strong classifiers, instead of many weak classifiers.

In practice, for each part, $\boldsymbol{\mu}_p$ is estimated and all the examples are projected in $T_{\boldsymbol{\mu}_p}$. The mapping of points on the Riemannian manifold to $T_{\boldsymbol{\mu}_p}$ and vice versa is done using the $\log_{\boldsymbol{\mu}_p}$ and $\exp_{\boldsymbol{\mu}_p}$ operators, respectively, as in [TPM08]. This mapping is done once, because all the following reasoning are done on $T_{\boldsymbol{\mu}_p}$ directly. The projected training examples are defined as $\{\boldsymbol{\Sigma}_{ip}, y_i\}_{i=1,\ldots,N}$, with $\boldsymbol{\Sigma}_{ip} \in T_{\boldsymbol{\mu}_p}$ and labels $y_i \in \{1 = \text{human part}, 0 = \text{other}\}$.

Using the binary LogitBoost algorithm [FHT00], a response function

$$F_p(\boldsymbol{\Sigma}_{ip}) : T_{\boldsymbol{\mu}_p} \mapsto \{0, 1\}$$

is estimated. It divides the tangent space into 2 parts, according to the training set of labelled items. This function, the strong classifier, is defined as a sum of weak classifiers. The probability of $\boldsymbol{\Sigma}_{ip}$ being in class 1 is represented by

$$P(\boldsymbol{\Sigma}_{ip}) = \frac{e^{F_p(\boldsymbol{\Sigma}_{ip})}}{e^{F_p(\boldsymbol{\Sigma}_{ip})} + e^{-F_p(\boldsymbol{\Sigma}_{ip})}} \quad F_p(\boldsymbol{\Sigma}_{ip}) = \sum_{l=1}^{N_l} f_l(\boldsymbol{\Sigma}_{ip}), \tag{5.14}$$

where $\{f_l\}_{l=1,\ldots,N_l}$ is the iteratively selected set of WLs. Each WL is estimated by solving a weighted least-square regression problem:

$$f_l = \sum_{i=1}^{N} w_{ip} |z_{ip} - f(\boldsymbol{\Sigma}_{ip})|^2, \tag{5.15}$$

where z_{ip} and w_{ip} denote the response values and the weights, respectively, in the following forms:

$$z_{ip} = \frac{y_{ip} - P(\boldsymbol{\Sigma}_{ip})}{P(\boldsymbol{\Sigma}_{ip}) - (1 - P(\boldsymbol{\Sigma}_{ip}))}, \tag{5.16}$$

$$w_{ip} = P(\boldsymbol{\Sigma}_{ip}) - (1 - P(\boldsymbol{\Sigma}_{ip})). \tag{5.17}$$

As regressors, second degree polynomial functions with no mixed terms are employed. This is because I experimentally found that this class of regressors is a good compromise between classification accuracy and computational complexity. In fact, the multidimensional polynomial fitting can be formalized as a linear problem [MSD97], and the complexity grows linearly with the number of terms used to solve the linear system. The second degree polynomial functions double the complexity with respect to the linear case, but the classification performances are clearly increased (see Fig. 5.9).

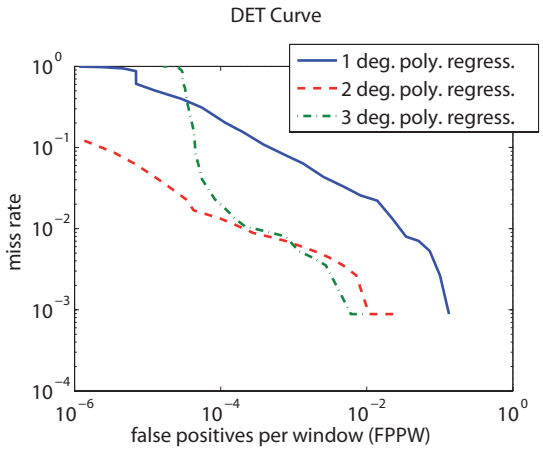


Fig. 5.9. Performances in terms of DET curve employing different regressor models – linear, second degree polynomial, third degree polynomial – in the proposed framework. The best performances are obtained with the second degree polynomial.

Part rejection cascade. The LogitBoost classifier of each body part is combined with a *rejection cascade structure* of K levels. Using a cascade makes the part detectors more robust to false positives. $N_n = 10^4$ negative examples are

sampled for each cascade level and joined to the N_p positive examples to form a training set of $N = N_p + N_n$ elements. F_p is rewritten in F_p^k to emphasize the dependence of the classifier on the current level.

All the negative examples are classified with the cascade of the previous $k-1$ classifiers, where $k \in \{2, \ldots, K\}$. The examples that are correctly classified (i.e. classified as negative) are removed from the training set, nevertheless keeping at least 1000 examples.

Training on cascade level k stops if a combined condition is satisfied. One would impose that the learning process correctly classifies at least 99.8% of the positive examples, and that it rejects at least 35% of the negatives. To verify this condition the dataset is sorted according to descending probabilities (Eq. (5.14)). Then, one can check that $F_p^k(\boldsymbol{\Sigma}_{ip}) > 0$ for at least the 99.8% of positives and $F_p^k(\boldsymbol{\Sigma}_{ip}) < 0$ for at least the 35% of negatives. The F_p^k value of the $(0.35N_n)$-th element with the smallest probability, denoted as $thrd_p^k$, is used for testing: a point $\boldsymbol{\Sigma}_{ip}$ is classified as positive if $F_p^k(\boldsymbol{\Sigma}_{ip}) - thrk_p^k > 0$.

5.3.1.2 Phase 2: Combination of part classifier

When the robust part classifiers are trained, their strong responses are combined into a unique human detection as follows:

$$F_{\text{comb}}(I_W) = \sum_{p=1}^{11} w_p \cdot F_p^*(\boldsymbol{\Sigma}_p), \tag{5.18}$$

where I_W is the detection window, $\boldsymbol{\Sigma}_p$ is the covariance matrix descriptor estimated on the body part p (projected into $T_{\boldsymbol{\mu}_p}$), and F_p^* is the classification response produced by the rejection cascade. I_W is classified as positive if $F_{\text{comb}}(I_W) > \tau$.

Since the location of the human body parts is fixed by construction and the variability of human postures is high, it is reasonable that some part detectors are more reliable than others. This is why a weight w_p is associated with each part classifier. Given a set of positive images, a validation dataset is instantiated, where the number of correct detections per part is counted. The normalized resulting values become the w_ps. w_p is proportional to the ability of F_p^* to classify its respective body part correctly, and it says which part is more suitable for the detection of human bodies.

5.3.2 Experimental Results

The proposed approach is evaluated considering the INRIA Person dataset [Dal05]. The dataset is not well-suited for training part-based classifiers (even if [YTCC09] uses it for the same purpose), because the data is not aligned, and different poses are present. This fact and the excellent results gained by the proposed approach witness the capability of the part-based classifier to absorb even strong pose variations.

The proposed framework is implemented with Matlab on an Intel Xeon 2.83 Ghz processor with 4.00 Gbytes of RAM. The training of the classifiers takes 15 minutes to generate a part-based classifier, for all the 11 parts, with at most 5 weak

classifiers per level. The state-of-the-art method in [TPM08] needs more than two weeks in the same hardware setting.

In Fig. 5.10, the proposed framework is compared with [TPM08] and the methods in [VJ02, DT05, DTTB07, SM07, MBM08], whose statistics are extracted from [DWSP09]. The performances are evaluated by adopting the Detection Error Tradeoff (DET) curve, that expresses the proportion of true detections against the proportion of false positives on a log-log scale. As visible from the results, the

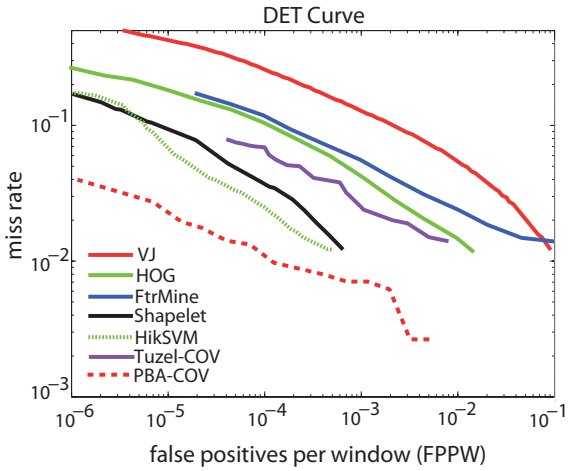


Fig. 5.10. Comparison with the state-of-the methods on INRIA Person dataset. The curves for other approaches are generated from [DWSP09] and [TPM08]. The proposed approach is named PBA-COV.

proposed framework outperforms all the other methods reaching the best performances, both considering the FPPW (False Positive Per Window) rate and the miss rate. Moreover, this holds in a boosting framework with very few WLs (for example, [TPM08] has 50% more WLs).

In Fig. 5.11 a comparison on the INRIA Persion dataset between the approach described in Sec. 5.2 and the approach presented in this Section is made.

Tab. 5.2 shows the ability of the proposed system to detect human body parts. The table is built by considering the cascade level $k = 5$. Considering that in the INRIA Person dataset, several people are not aligned. However the proposed part detector is able to detect single parts with high accuracy. In specific, the more reliable region is the torso, meaning that the part descriptor is particularly suited for that body portion, capturing all the pedestrian intra-class variability. Such variability pops out considering the variance image of the INRIA training dataset (see Fig. 5.12), where in each pixel the associated per-pixel variance is portrayed. It is evident that, even if that portion is characterized by the highest variability, it is the best modelled by the proposed framework. The weights used to build the final detection response are proportional to the number contained in Tab. 5.2.

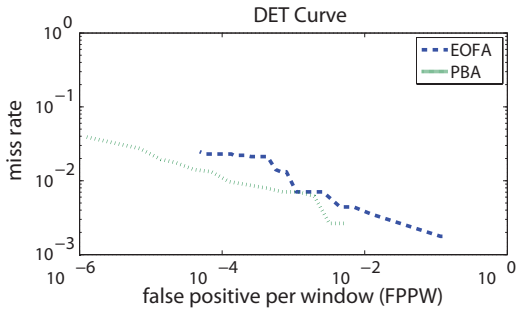


Fig. 5.11. Comparison between the FUD approach described in Sec. 5.2 and the approach presented in this Section (Sec. 5.3) named PBA.

Human body part	Accu.% ($k = 5$)
full body	99.4%
head-shoulder	93.8%
torso	97.2%
legs	92.8%
left shoulder	82.7%
head	83.6%
right shoulder	87.4%
left arm	88.6%
right arm	88.4%
left leg	81.8%
right leg	84.9%

Table 5.2. Per-part detection accuracy. The detection ability of the part detectors in the cascade level $k = 5$ is shown.

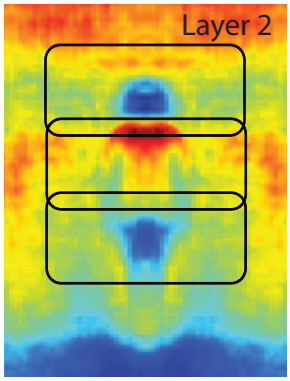


Fig. 5.12. Capturing the intra-class variation: the central part, even if characterized by the highest variance, is the best detected by the part classifier.

5.4 Low Resolution Pedestrian Detection via $\mathbf{SST_{struct}}$ Tensors

This Section concerns with the pedestrian detection task where the human body covers a smaller portion of the image to be detected, that means it is visible at lower resolutions. This covers outdoor settings such as for the surveillance case. Low resolution pedestrian detection is a difficult task from a computer vision point of view. The absence of explicit models leads to the use of discriminative learning techniques, where an implicit representation is learned from examples.

Here, the usage $\mathrm{SST}_{\mathrm{struct}}$ tensors is proposed to measures the self-similarity of the parts of a human body and use this source of information as a feature. Therefore, since parts are tricky to be modelled from low resolution images, a pyramidal regular grid of patches [LSP06] is adopted. With this settings $\mathrm{SST}_{\mathrm{struct}}$ beats the COV tensor representation for the low resolution pedestrian detection task on the DaimlerChrysler [MG06] pedestrian detection benchmark.

5.4.1 Object Model for Low Resolution Pedestrians

In order to model complex objects like pedestrians, a pyramidal representation is adopted. To build robust descriptor, one can follow the idea proposed in [LSP06, BZM07a], where a pyramidal patch based representation is used. In particular, each image is divided into a sequence of increasingly finer spatial grids by repeatedly doubling the number of divisions on each axis direction. A 3 level pyramid is adopted as depicted in Fig 5.13. $\mathrm{SST}_{\mathrm{struct}}$ is combined with pyrami-

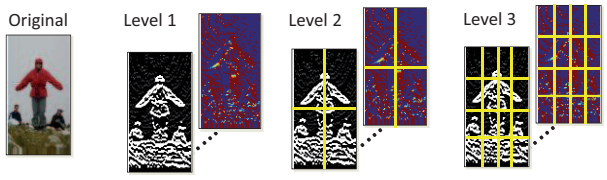


Fig. 5.13. Spatial pyramid representation. An image on the left and grids for levels 1 to 3.

dal structure, since that structure guarantees at the same time a high level of robustness and generality to describe different classes of objects. However, how to decide the patch size or rather the grid layout still remains a main issue. The hypothesis is that a rougher grid layout is suitable for a task like object detection in which the object model must be invariant (or at least less sensitive) to object details. Adopting a finer one, the task has to be necessarily changed into an object classification task with a higher level of details to discriminate among classes. The next experimental section confirms that hypothesis on the DaimlerChrysler [MG06] pedestrian detection benchmark.

5.4.2 Experimental Results

This Section contains an experimental study to use that representation for *small* pedestrian detection task in real scenarios. The DaimlerChrysler dataset [MG06] is chosen for this purpose because it contains very small pedestrians.

The DaimlerChrysler dataset [MG06] contains 4000 pedestrian (24000 with reflections and small shifts) and 25000 non-pedestrian images. The dataset was organized into three training and two test sets, each of them having 4800 positive and 5000 negative examples. The small size of the pedestrian windows (18×36 pixels), combined with a carefully arranged negative set, makes detection on the DaimlerChrysler dataset extremely challenging.

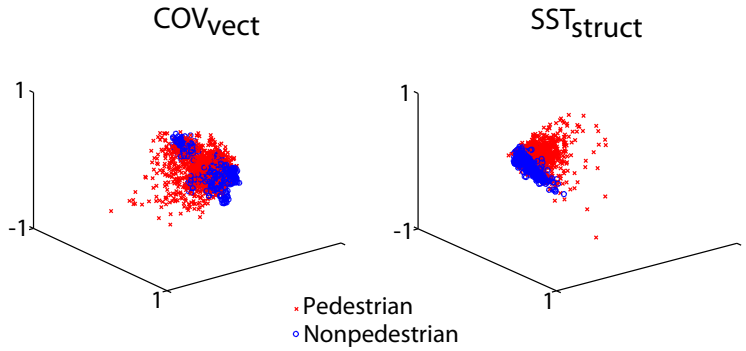


Fig. 5.14. DaimlerChrysler feature space visualization via PCA using $\mathrm{COV}_{\mathrm{vect}}$ and $\mathrm{SST}_{\mathrm{struct}}$.

For this dataset $\mathrm{SST}_{\mathrm{struct}}$ build upon a set of COV tensors and $\mathrm{COV}_{\mathrm{vect}}$, that simply concatenates all the COV tensors, are compared. Therefore COV tensors are the basic ingredient both for the representations to make that comparison as fair as possible. To extract COV tensors, each image is gridded extracting 8 patches using the covariance of gradient-based information for each patch. The color information is not considered since it is not available for this dataset. More formally, the feature set is:

$$\Phi(I,x,y) = \left[G_{|\,|}(I)\ G_O(I)\ D_x(I)\ D_y(I)\ D_{xx}(I)\ D_{yy}(I)\right], \tag{5.19}$$

where $G_{|\,|}(Y)$ and $G_O(Y)$ are the gradient magnitude and orientation, and $D_x(I)$, $D_y(I), \ldots$ are intensity derivatives. Then a covariance matrix is computed for each image patch using the feature set above. Covariances are vectorized and used as feature descriptors. Then $\mathrm{SST}_{\mathrm{struct}}$ is built computing the distance between each pair of descriptors as formalized in Eq. (4.2), where d is the Euclidean distance. On the contrary, as mentioned above, $\mathrm{COV}_{\mathrm{vect}}$ is built concatenating all the vectorized covariance matrices.

In Fig. 5.14 PCA (Principal Component Analysis) is applied to visualize the distribution of the negative and positive sets using the two different representations. One can observe that $\mathrm{SST}_{\mathrm{struct}}$ offers a more linearly separable feature space regarding $\mathrm{COV}_{\mathrm{vect}}$. Hence, one may expect that the detection performances of $\mathrm{SST}_{\mathrm{struct}}$ are reasonably better than $\mathrm{COV}_{\mathrm{vect}}$.

Fig. 5.15 shows another experiment in order to evaluate the behaviour of $\mathrm{SST}_{\mathrm{struct}}$ at different patches resolution. For this Figure, a pyramidal $\mathrm{SST}_{\mathrm{struct}}$

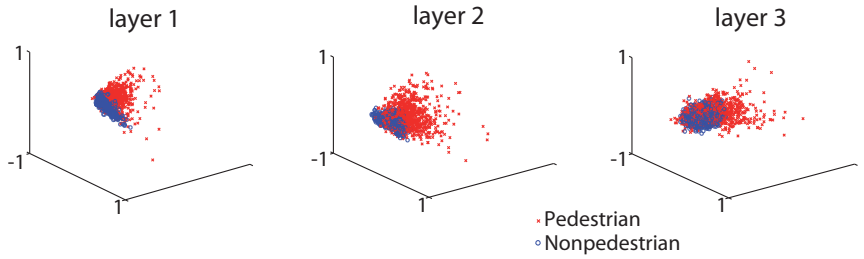


Fig. 5.15. DaimlerChrysler $\mathrm{SST}_{\mathrm{struct}}$ feature space visualization via PCA at different patch size.

is built dividing an image into a sequence of increasingly finer spatial grids by repeatedly doubling the number of divisions on each axis direction. Therefore, Fig. 5.15 depicts the feature space for different layers of the spatial pyramid. For each level of that pyramid an $\mathrm{SST}_{\mathrm{struct}}$ is computed and the feature space associated with each pyramid layer is visualized. One can observe that a rough grid is more suitable for the detection task, while a finer grid subdivision can be used for a different classification task in which a high level of details is necessary (e.g. pose classification). To verify my assertion, the performances of the different pyramid layers for the pedestrian detection task are compared. In Fig. 5.16(a), the DET curve is plotted on a log-log scale, whose y-axis corresponds to the miss rate, and the x-axis corresponds to false positives per window (FPPW). One may notice that the first (top) layer is the most indicated for the detection task because its rough image subdivision captures only the essential information to characterize an object avoiding its details which are unnecessary for the detection task. In Fig. 5.16(b) shows how adding the spatial layout (i.e. concatenating the x and y coordinates of each patch) and an appearance prior to the feature descriptors the detection performances can be increased. The appearance prior is injected into the $\mathrm{SST}_{\mathrm{struct}}$ as a probabilistic map (where 1 means a pixel that certainly belong to a pedestrian) added to Φ (5.19) and computed using the generative Steal model [JPC+09]. Finally, in Fig. 5.16(c) is shown the effect of changing the metric used to compare COV tensors and, as predictable, using the Riemannian metric instead of the Eucledian one the the performance of the framework improve.

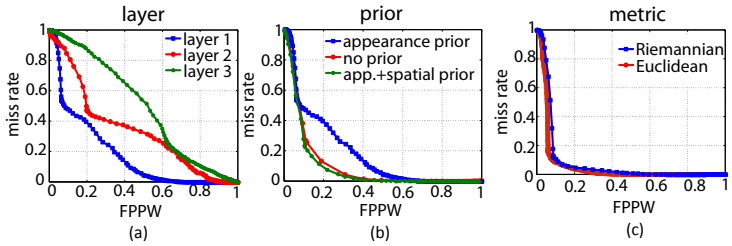


Fig. 5.16. DET curve on the DaimlerChrysler dataset using the $\mathrm{SST}_{\mathrm{struct}}$ tensor. (a) depicts the detection performances associated with different levels of the spatial pyramid. (b) shows how adding the spatial layout and an appearance prior to the feature descriptors the detection performances can be increased. (c) compares two different metrics that can be used to build the $\mathrm{SST}_{\mathrm{struct}}$.

5.5 Robust Pedestrian Detection using Hausdorff Distance

The capability to detect people in images of crowded scenes is fundamental for a large variety of applications, such as video surveillance or automatic driver-assistance systems. If people detection is performed in a non-problematic scenario, such as one where people are not occluded, with a limited range of scales and pose variations, there are already a lot of effective frameworks [DT05, TPM06, GL09, DTPB09, FGMR10] usable to solve this task. On the other hand, if the scenario is problematic, among these systems only few are really useful. It is worth noting that three of them which are able to manage different difficult problems which are typically present jointly in images of crowded scenes. [MG06] effectively deals with small scale pedestrians, [LSS05] manages the presence of occlusions and [FGMR10] covers extreme changes of pose or occlusions of pedestrians. Since anyone of the previous frameworks is able to give a solution to all the above-mentioned problems, this Section proposes a unified framework capable to jointly cope with the described issues. Hence, the goal is to detect as many people as possible even when the human body layout cannot be inferred.

I propose to replace the definition of a person as a set of fixed parts as a set of non-fixed combination of human patches which share a defined space location in the image. Initially, an image is divided into a set of multi-scale overlapping patches on which a binary patch classifier is learned in order to highlight the patches belongings to people. Then the human patches are assigned, if it is possible, to the different people in the image.

The ideas below the proposed approach are: 1) a person is represented as a variable set of patches depending on a probabilistic evaluation of the patch visibility, or rather if a human is occluded the patches containing the occlusion are automatically removed from the model. 2) since the number of patches is variable, a classifier based on a set distance is used to discriminate between human and nonhuman image ROIs (Regions of Interest).

5.5.1 The Approach

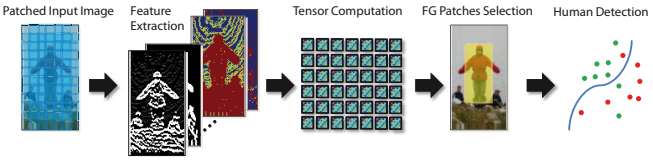


Fig. 5.17. The proposed approach pipeline.

The proposed approach is a five-phase process. (1) A set of features is calculated on a set of overlapping patches for each image. In the training stage, it has a set of ROIs containing fully-visible people at the same scale. Then the training set is populated by other problematic examples where occlusions are present. (2) From the training ROIs a set of features is extracted, and (3) extracting a fixed number of patches computed on a regular grid, their tensor descriptor is computed. (4) A robust binary patch classifier is used to detect the foreground (human) patches. (5) The survived patches are organized as sets and, using a classifier based on a set distance, one can finally detect the presence of a human in the original ROI. The set distance is necessary since the number of foreground patches is variable. In Fig. 5.17 the entire approach pipeline is depicted.

5.5.1.1 Person Representation

For the pedestrian detection task the most reliable source of information is related to the image gradient. As shown in [DWSP11], that information is strictly dependent on the image resolution. In particular, for low resolution pedestrians (less than 30 pixels tall) Haar Wavelet features [MG06] are a simple and effective choice, while for medium and high resolution pedestrians it is preferable to use directly the gradient information or its orientation as done by HOG (Histogram of Oriented Gradients) [DT05]. To be able to manage people at different resolutions, combinations of the previous features are used [DTPB09]. This combination is typically a straightforward concatenation among some of the previous features. This leads to two problems: 1) using different features, the normalization is not an easy task and it becomes more difficult proportionally to the number of the features involved. 2) The dimension of the final vector representation can be extremely high, causing the curse of dimensionality problem. A more proper way to combine different features and automatically solving these problems is using covariance tensors as feature descriptors [TPM06]. Due to the use of integral representation, these descriptors are fast to compute, making it suitable for detection tasks. It has shown that there are other tensor representations (see Chap. 4) able to outperform the covariance, but their calculation time is still too expensive for object detection purposes.

Regular Grid Human Body Layout. It is worth noting that it is necessary to make a step further to the definition of human body part widely used for the

current pedestrian detectors [FGMR10] in order to find a good representation for a person in a crowded scene where small pedestrians are present. This is because 1) a configuration of body parts changes accordingly to the object resolution. Even if multiple models are instantiated (one for each object resolution), their management could be tricky and computationally expensive. 2) The part alignment problem is automatically involved in the process of defining it. Since the part configuration can vary slightly with highly non-rigid objects (as a human) or in case of occlusion, the research of the correct position and scale could lead to very poor results. 3) Parts are extremely unusable descriptors in crowded situations where it is hard to assign parts to different overlapped human bodies correctly.

The proposal is to divide an image I in overlapping patches on a regular grid. Each patch is described by a COV tensor. More formally, a set of patches $\{P_i\}_{i=1,\ldots,N}$ of 4×4 pixels is sampled from I as shown in Fig. 5.17. Unlike many successful people detectors [TPM08, DTPB09], in this case the patch dimension p is not optimized in order to obtain the best performance on a benchmark dataset. This should be led to a more general detector in which the concept of fixed human parts is replaced by one that describes it as variable human patches.

5.5.1.2 Combinations of Features

Each patch P_i is represented by a covariance matrix of d image features

$$\Phi = [H_1 \; H_2 \ldots H_{10} \; G \; O] \,, \tag{5.20}$$

where d is equal to 12. $H_1, \ldots, H_{10}$ represent the results of the application of the set of Haar Wavelets. the variances of the defined features and their correlations with each other, which are useful to detect both high and low resolution people. In order to build a set of covariance matrices quickly, given a set of feature Φ, in [TPM06] a good solution, based on the integral representation which is adopted in this work, is proposed.

Given a set of $d \times d$ covariance descriptors $\{\mathbf{C}_i\}_{i=1,\ldots,N}$ where $\mathbf{C}_i \in Sym_d^+$ (the group of the symmetric positive definite matrices), they are one-to-one with their relative patches $P_1, \ldots, P_N$. A very important preprocessing operation is the normalization of these descriptors to enhance the robustness to include also illumination variations in I. Unlike the local normalization in [TPM08], the idea is to use a global normalization which is much more robust in the presence of occlusions and noise. The normalized version of a covariance matrix $\mathbf{C}_i$ is denoted as $\hat{\mathbf{C}}_i$ and is computed by dividing columns and rows of $\mathbf{C}_i$ with the square root of the maximum variance of the image features Φ (Eq. (5.20)):

$$\hat{\mathbf{C}}_i = \mathrm{diag}(\mathbf{V})^{-\frac{1}{2}} \mathbf{C}_i \, \mathrm{diag}(\mathbf{V})^{-\frac{1}{2}}, \tag{5.21}$$

where $\mathrm{diag}(\mathbf{V})$ is a diagonal matrix in which there is the maximum variance of the image features at the diagonal entries. This is equivalent to first globally normalizing the feature vectors to have zero mean and unit standard deviation and then computing the covariance descriptor.

Covariance matrices are an interesting way to combine information also owing to their particular geometry, which provides an implicit framework to represent

multi-modal distributions. consequently, if the focus is on a sub-set of covariances (i.e. people patches), a set of tools is naturally provided to find a highly discriminative Euclidean space to analyse them exploiting their geometry as described in the next section.

Covariance Tensors. Since covariance matrices do not live on a vector space, it is necessary to map them on a tangent space of this manifold (i.e. $T_{\mathbf{M}}Sym_d^+$) where the covariances can be treated as vectors. More formally, given a normalized covariance matrix $\hat{\mathbf{C}}_i$ it can be projected applying the following equation which represents the logarithmic mapping

$$\mathbf{c}_i = \mathbf{M}^{\frac{1}{2}} \log_{\mathbf{I}_d}(\mathbf{M}^{-\frac{1}{2}} \hat{\mathbf{C}}_i \mathbf{M}^{-\frac{1}{2}}) \mathbf{M}^{\frac{1}{2}}, \tag{5.22}$$

where $\mathbf{M} \in Sym_d^+$ is the Karcher mean point computed considering only the covariances belonging to people image examples and is computed [Kar77]. The $\log_{\mathbf{I}_d}(\mathbf{A})$ map is equal to $\mathbf{U}\log(\mathbf{D})\mathbf{U}^T$, where $\mathbf{UDU}^T$ is the eigenvalue decomposition of $\mathbf{A}$. It should be noted that $\log_{\mathbf{I}_d}$ and log are different operators. The first one is a standard operator of the Riemannian geometry and the second one is the usual logarithm of a scalar value (for further details see [TFC+10]).

Considering that $\mathbf{c}_i \in Sym_d$, it contains only $d(d+1)/2$ independent coefficients which can be the upper triangular part of the matrix. As in [TPM08], an orthonormal coordinate system for the tangent space is defined as in Sec. 2.2.6.

Having $d = 12$, a tangent vector is a 78 dimensional. Since not all the features are informative, linear PCA (Principal Component Analysis) is applied. According to [ZLY10] the 96% of the energy is preserved selecting the principal components, which number is automatically selected. The principal components vector after the projection is denoted with $\tilde{\mathbf{c}}_i$

$$\tilde{\mathbf{c}}_i = \mathbf{T}\mathbf{c}_i, \ \mathbf{T} \in \mathbb{R}^{d(d+1)/2 \times d_p} \tag{5.23}$$

where $\mathbf{T}$ is learnt during the training phase and d_p is automatically selected. As done above for the patch dimension, the goal is to find the best feature set Φ to obtain the best performance on a benchmark dataset. One should collect a reasonable feature set that can be used to describe pedestrians at different scales and to use PCA [ZLY10] to select automatically the most informative subset of the original covariance $\mathbf{C}_i$.

A further dimension is added to $\tilde{\mathbf{c}}_i$, and it contains a rough spatial information position in order to avoid patch configuration clearly infeasible. Dividing the ROIs in 3 equal horizontal layers 1, 0 and -1 is assigned to the *top*, to the *middle* and to the *bottom* body part respectively.

5.5.1.3 Patch Classification

A large number of human and non-human patches is collected and binary classifier is trained using RF (Random Forest). $P(\tilde{\mathbf{c}}_i)$ is the probability of a patch to belong to a human. That probability is computed as

$$P(\tilde{\mathbf{c}}_i) = \frac{1}{T_n} \sum_{t=1}^{T_n} g_t(\tilde{\mathbf{c}}_i), \tag{5.24}$$

where T_n is the cardinality of the trees and $g_t(\tilde{\mathbf{c}}_i)$ is a decision function given by the t-th tree. Hence, $P(\tilde{\mathbf{c}}_i)$ is computed as the mean of the decision responses coming from all the decision trees. Finally, if $P(\tilde{\mathbf{c}}_i) > .5$ then $\tilde{\mathbf{c}}_i$ is associated with a human patch. Clearly, one cannot expect that this classifier is accurate, since extracting small patches the human and non-human classes have a large overlap. This is actually the reason why RF is chosen as classifier: it is able to manage very noisy data and to find a rough subdivision that removes non-human patches.

5.5.1.4 Object Detection based on Hausdorff distance

After the previous pruning phase one can await to have a reliable set of patches for each example in the training set. Then, a high-level classifier is built and it should be able to manage a variable representation of the same object to label a ROI as a pedestrian. So, the feature descriptors of the survived patches are treated independently, so that the descriptors are not concatenated in a unique vector the order among the patches is lost if some patches are removed. Moreover, standard machine learning techniques cannot manage representation of different dimensionality. A popular distance among two sets of points, that works regardless the number of descriptors in each set, is the Hausdorff distance. It has already been used for object recognition in quite recent works [DJ94, Fel01], but in these cases object descriptions were image coordinates. Since the space on which the features lie is $\mathbb{R}^n$ $(n = d(d+1)/2)$, it is possible generalize the usual Hausdorff distance using the Euclidean norm of $\mathbb{R}^n$. Therefore, to compute the Hausdorff distance of a pair of descriptor sets $\tilde{\mathbf{C}}_1, \tilde{\mathbf{C}}_2$ one may proceed as follows:

$$d_H(\tilde{\mathbf{C}}_1, \tilde{\mathbf{C}}_2) = \max[\max_{\tilde{\mathbf{c}}_i \in \tilde{\mathbf{C}}_1} (\min_{\tilde{\mathbf{c}}_j \in \tilde{C}_2} (||\tilde{\mathbf{c}}_i, \tilde{\mathbf{c}}_j||)), \max_{\tilde{\mathbf{c}}_j \in \tilde{\mathbf{C}}_2} (\min_{\tilde{\mathbf{c}}_i \in \tilde{C}_1} (||\tilde{\mathbf{c}}_j, \tilde{\mathbf{c}}_i||))] \quad \tilde{\mathbf{c}}_i, \tilde{\mathbf{c}}_j \in \mathbb{R}^n. \tag{5.25}$$

The Euclidean norm is chosen for computational convenience, but any norm of $\mathbb{R}^n$ can be used to into Eq. (5.25). Than d_H is embedded into an $\mathrm{SST}_{\mathrm{struct}}$ (see Sec. 4.2.3) computed on the training set denoted as D. After that a kernel matrix is built exploiting D. Since D cannot satisfy the Mercer inequality itself, to build a valid kernel that can be combined with an SVM the non-linear transformation described in Sec. 3.4.2 is applied. Applying that transformation the Mercer inequality is satisfied, hence the D^+ is a valid kernel. In Fig. 5.18 an example of the kernel matrix based on d_H is shown.

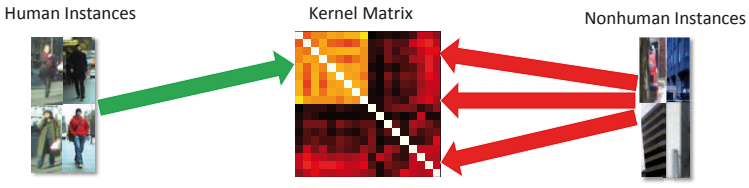


Fig. 5.18. An example of Kernel matrix based on the proposed Hausdorff distance.

Once the kernel is built, a binary SVM is trained for the final pedestrian detection task.

5.5.2 Experimental Results

In the first experiment, the probabilistic output of the patch classifier described in Sec. 5.5.1.3 is shown in the presence of different types of synthetic occlusions. The goal is to find a reliable set of patches that can be used to describe a human. The result of the application of different kinds of occlusions are shown in Fig. 5.19. One may notice that, although the grid of image patches is quite rough, the patches classifier provides useful information on which is the actual object ROI for each occluded image. One can object that the segmentation should be finer, but for detection purposes it is necessary to minimize the computational burden, therefore a rough image segmentation is enough for this first pruning phase. In the next

Fig. 5.19. Patch classification in the presence of different types of synthetic occlusions. Each picture shows the occluded image (on the left) and the image patches classification that produces a probabilistic map (on the right). Different levels of occlusion are randomly adopted: from soft (25% of the image size) to hard (50% of the image size). Various kinds of noise are also tried: full occlusion and salt& pepper noise.

experiment, regarding again the output of the patch classifier (Sec. 5.5.1.3), the probabilistic map produced by the patch classifier is shown in function of the image resolution. It is interesting to observe that the final probabilistic map is still reliable even when the original occluded image is heavily downscaled. That means two things: 1) the patch classifier can provide reliable information also in presence of heavy noise and low resolution images, 2) the feature set adopted (see Eq. (5.20)) is effective, so it captures discriminating information in very low resolution images.

The proposed framework is trained exploiting the INRIA dataset [Dal05]. The dataset is partitioned into two, where 2416 pedestrian annotations and 1218 non-pedestrian images, from which 100000 non-pedestrian ROIs of 64×128 pixels, are extracted. The remaining images compose the testing set. Since that dataset does not include low resolution pedestrians, the proposed framework is tested on the images of the Caltech pedestrian dataset [DWSP11], which contains several images with both very low and hight resolution pedestrians in urban scenarios. In Fig. 5.21 some qualitative results are depicted. The proposed method achieves

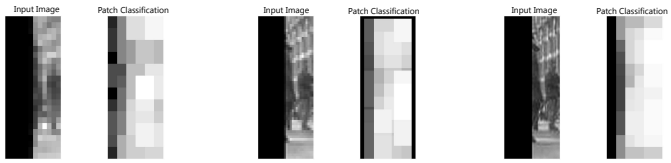


Fig. 5.20. Patch classification at different image resolutions. The full resolution image on the right. It is downscaled one time to obtain the central image and two times for the left image. Each image presents two maps: on the left the occluded image and on the right the image patches classification that produces a probabilistic map.

good performances in the pedestrian detection where the pedestrians are small. The number of false alarm is low, but many pedestrians are lost.

Fig. 5.21. Detection examples. The classifier is trained on the INRIA dataset [DT05]. Red boxes all the detection results without filtering or maximum suppression. In the first two rows there are good detection examples considering medium and low resolution pedestrians. Problematic detection images are shown in the last row.

Discussion. There are two main issues that must be tackled in order to improve the performance of the proposed detection approach. The first issue regards the efficiency: in fact, the usage of kernel methods in detection problems is very limited due to its computational burden. Since the patch detector permits to a considerable number of false positives to reach the kernel based classifier, it is difficult to build a light kernel that permits a fast detection. Therefore, it is nec-

essary to improve the performance of the patch classifier using contextual and spatial information during the pruning phase.

Another issue concerns the Hausdorff distance. That distance assumes that the information contained into the descriptor vectors is geometrical, namely vectors, should contain coordinates of a $1, 2, \ldots, N$ dimensional space. For detection purpose the descriptors contain different kind of information. That leads to an unclear meaning of that distance from the geometrical point of view. However, the proposed distance is effective on the pedestrian detection task (see Fig. 5.18).

5.6 Embedded Object Detection

In Computer Vision, the object detection problem is a fundamental task, but only a few systems are thought to be realized on an embedded architecture. To this end, in this Section a highly-parallelizable classification framework for an FPGA-based implementation is designed, which is suitable for generic detection problems. This because in this case the OI parts layout is not designed specifically for a class of objects, in fact a regular grid of overlapped patches is used. Then, each patch is represented by the COV tensor of a small set of features explicitly selected for detection purposes. The model consists in a weighted sum of boosted binary classifiers, trained the set of overlapped image patches. The hardware design has been developed in parallel and with specific architectural solutions, allowing a fast response without degrading the functional performances. In this case, the weighted regression trees are adopted as basic classification tool to achieve both the best classification performances and the maximum robustness to noise. Weighted regression trees are an alternative method to non-linear regression, so they can be used by LogitBoost as weak learners.

In Sec. 5.6.1, the high level architecture and some details of the software implementation are presented. The hardware implementation design is reported in Sec. 5.6.2. An experimental study on the problem of the pedestrian detection is proposed in Sec. 5.6.3,

5.6.1 The Parallel Classification Framework

The proposed classification framework has been designed specifically to be implemented on a high parallelizable architecture. In fact, as shown in Fig. 5.22, the image containing the OIs is organized into a uniform sampled set of overlapping patches. For every patch, a COV tensor is independently built in order to describe it, and a binary or multi-class classifier is also applied for assigning a label.

5.6.1.1 Object Descriptors

Given a set of N_p patches, the corresponding set of covariance matrices is denoted as $\{\mathbf{C}_i\}_{i=1,\ldots,N_p} \in Sym_d^+$ (the space of symmetric positive definite $d \times d$ matrices), where d is the number of features involved to build the matrices. For the sake of clarity, here a set of clues which will be fed into the covariance matrices for their

usage in the experiments is instantiated. Likewise [TPM08], for each pixel (x, y) inside the patch, the following source of information is extracted:

$$[|I_x| \ |I_y| \ \sqrt{I_x^2 + I_y^2} \ |I_{xx}| \ |I_{yy}| \ \arctan \frac{I_x}{I_y} \ ]^T, \tag{5.26}$$

where I_x, I_{xx}, etc. are grey-level intensity derivatives, and the last term represents the edge orientation. From the features vector in Eq. (5.26), a 6 ($d = 6$) covariance matrix can be estimated. In order to disregard the expensive computation and the complex management of geodesic distances among COV tensors, it is recommended to project the covariance matrices form their Riemannian manifold $\mathcal{M}$ to at least one of the tangent spaces of $\mathcal{M}$.

By computing the sectional curvature of $\mathcal{M}$ [Cha06] (i.e. the natural generalization of the classical Gaussian curvature for surfaces), it is possible to show that this space is almost flat (this is demonstrated in Chap. 6 with great detail). This means that the neighbourhood relation between the points on $\mathcal{M}$ remains unchanged, wherever the projection point is located.

Therefore, the most convenient projection point from the computational perspective is the $d \times d$ identity matrix $\mathbf{I}_d \in \mathcal{M}$. The projection translates the covariances into $\{\mathbf{c}_i\}_{i=1,\ldots,N_p}$ vector descriptors, such that $\mathbf{c}_i \in \mathbb{R}^{d\cdot(d+1)/2}$. More precisely, this projection is called *logarithmic mapping* and it is a standard Riemannian geometry operator which provides a linearised version of $\mathcal{M}$.

In practice, the logarithmic mapping is formulated using the EVD decomposition (see Sec. 2.2.5 for details) as

$$\log_{\mathbf{I}_d}(\mathbf{X}) = \log(\mathbf{X}) = \mathbf{U} \log(\mathbf{D}) \mathbf{U}^T. \tag{5.27}$$

Moreover, the tangent space is Sym_d, where there are only $d(d+1)/2$ independent

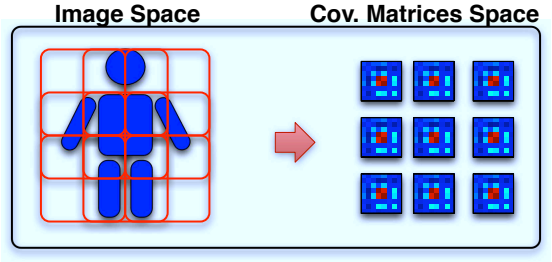


Fig. 5.22. The patch-based model. Considering the human body as an OI, it is subdivided into 9 overlapped patches and each patch is described by a covariance matrix descriptor.

coefficients, which are the upper triangular or lower triangular part of the matrix, by applying the *vector* operator (defined in Sec. 2.2.6). It relates the Riemannian metric on the tangent space to the canonical metric defined in $\mathbb{R}^{d(d+1)/2}$.

5.6.1.2 Object Classification

Considering the previous set of patches

$$\{\mathbf{C}_i\}_{i=1,\ldots,N_p}$$

, a set of classifiers

$$\{F_i\}_{i=1,\ldots,N_p}$$

is trained, one for each patch, through the LogitBoost algorithm [FHT00], adapted to work on Riemannian manifolds. This version differs from the original one for a pre-processing step in which Eq. (5.27) and (2.1) are applied to the entire training set (see Alg. 6 for the schematic of this procedure). When the patch classifiers are learnt, their strong responses are combined into a unique classification response as follows:

$$F_{\text{comb}}(I_W) = \sum_{p=1}^{N_p} w_p \cdot F_p(\mathbf{c}_p), \tag{5.28}$$

where I_W ($\subset I$) is the detection window (of an image I), which is classified according with the sign of $F_{\text{comb}}(I_W)$: if it is positive, I_W belongs to the current class. However, to achieve the best classification performances, a more restrictive condition is imposed, i.e. $F_{\text{comb}}(I_W) > \tau$, where the parameter τ depends on the specific application. Since the location of the OI patches is fixed by construction and their variability could be high, it is reasonable that some patch detectors are more reliable than others. This is why a weight w_p is associated to each patch classifier.

Given a set of OI examples, a validation dataset is instantiate, where the number of correct classifications per patch is counted. The normalized resulting values become the weights w_ps to be used in the testing phase. Therefore, each w_p is proportional to the ability of F_p to classify its associated patch correctly, and says which patch is more suitable for the classification task in general.

5.6.1.3 Weak Classification Strategy

It is possible to use very different types of WLs for boosting purposes. The most common are the decision stumps (or regression stumps), which are piecewise constant regression functions or linear regression functions. To address both binary and multi-class classification problems, the best weak classification strategy is represented by the weighted *regression trees* [Bre84]. In order to avoid the risk of overtraining of the regression tree, a minimal number η of observations per tree leaf is experimentally estimated.

5.6.1.4 Object Model Learning

It is necessary to define some fundamental details to build the OI model. First, it is important to specify an automatic stop criterion for the training phase. The proposed rule is a composition of two terms. The first one takes into account the accuracy with which the problem classes are correctly classified by setting the maximum accuracy τ_{acc} for all the classes. The second one concerns the *learning rate*, which is the difference in accuracy between two consecutive iterations of LogitBoost. If the learning rate is less than τ_{lr} for all the classes, then the boosting process has converged to its optimal solution. In the experiments, τ_{acc} is set to 99% and τ_{lr} to 1% to obtain the best classification performances.

5.6.2 Implementation Design

The system described in Sec. 5.6.1 and illustrated in Fig. 5.23 has been designed to largely fit on a FPGA, specifically, a Xilinx Spartan-3A DSP 3400A device, which has 23872 slices and 126 hardware multipliers. This Section aims to briefly introduces the hardware implementation design of the proposed architecture using a limited amount of resources in order to be integrated in an existing circuit with extended functionalities.

An overview of the proposed architecture is shown in Fig. 5.23. Its design consists in a pipeline composed of five main stages which can be grouped into three main architectural components. The first one is the *Covariance Matrix Computation* module, which calculates the covariance matrix $\{\mathbf{C}_i\}_{i=1,\dots,N_p}$ of each patch by extracting the feature vector for each incoming pixel. Then, the *Logarithm Projection* module targets at projecting the covariance matrices on a tangent space in order to classify the patches. The logarithmic projection is applied according to Eq. (5.27), which requires an eigenvalue decomposition of covariance matrices, followed by the computation of the eigenvalue logarithm.

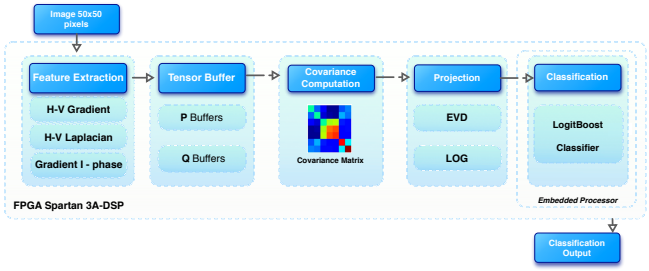


Fig. 5.23. General scheme of the architecture.

Since the goal is to minimize the use of hardware resources while slightly degrading the throughput, the eigenvalue decomposition module has been designed using the same single processor instead of a systolic architecture. This has been possible by adopting an iterative Jacoby-like method [GVL96], and exploiting the sequential property of each iteration.

The last stage is the *Classification* module, which aims to classify the covariance descriptor by using weighted regression trees.

The hardware implementation details of all the modules go beyond the goal of this thesis, but can be found in [MTF+10].

5.6.3 Experimental Results

For the validation of the proposed classifier, a software-based classifier specialized in a binary classification problem is implemented. The challenging task of pedestrian detection is chosen to compare the results with the large number of competitors in the literature. The results of a preliminary accurate investigation

of the proposed hardware architecture are reported by using a software floating-point based on a behavioural model.

The INRIA Person dataset [Dal05] is considered for the validation of the classifier. The central region inside the pedestrian detection window (corresponding to the actual region where the pedestrian is enclosed) is picked. The region is divided exactly as depicted in Fig. 5.22. A covariance descriptor is associated with each

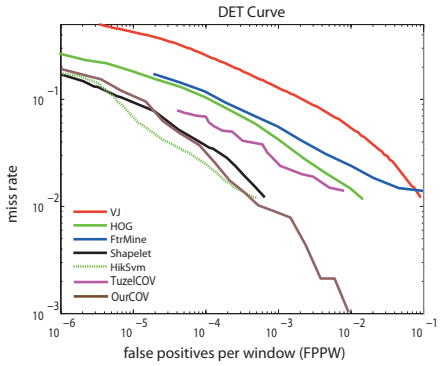


Fig. 5.24. DET curve for pedestrian detection: comparison of the proposed framework (ourCOV) with state-of-the-art methods.

patch, and it is estimated using the following procedure. For each patch, a binary classifier is built as described in Sec. 5.6.1. The η parameter, ruling the complexity of the regression trees, has been fixed on the optimal value 150. We use a rejection cascade [VJ01] of 5 levels in which each level is populated by 10000 background examples. Augmenting the number of cascade levels to more than 5 does not increase the accuracy appreciably, since the number of covariance features remains fixed (in [TPM08], instead, at each step a new set of features is selected).

In Fig. 5.24, the proposed framework is compared with the methods in [DT05, TPM08, VJ02, DTTB07, SM07, MBM08], whose plots are extracted from [DWSP09] and [TPM08]. The performances are evaluated using the Detection Error Tradeoff (DET) curve, that expresses the proportion of false negatives against the proportion of false positives, on a log-log scale. The curve is estimated by varying the threshold τ in the range $[-1, 1]$. The proposed detector defines state-of-the-art performances, especially in terms of miss-rate. Considering that, the framework has a extremely general and slim structure with respect to the state of the art, this is a particularly promising result.

So far, the real time performances are not achievable in software. The implementation of this classifier in hardware will allow to meet also the real time constraint. As described in Sec. 5.6.2, the eigenvalue decomposition module is the bottleneck of the design and determines the timing performance of the overall system. Considering the system speed, an excellent performance would be achieved, compared to other alternatives such as [AA04, BML$^+$08]. Particular efforts have been devoted during the design to the optimization of the Logarithm Mapping Module which represents the most challenging element to be implemented, de-

manding resource and being critical from the accuracy point of view. The approximation effect resulting from the porting of the classifier in hardware is evaluated by computing the relative error as the relative value difference of the Frobenius norm of two datasets, the set $\mathcal{M}$ of projected covariance matrices computed with the floating-point classification framework described above and the set of matrices projected using the Logarithm Mapping Module. The mean relative error is 0.2537%. The evaluation of the fixed-point accuracy of the Logarithm Mapping Module is promising and it allows to go further towards a real time embedded system for multi-class classification problems.

5.7 An Experimental Comparison for Video Surveillance

This Section shows the uses of some of the previous pedestrian detection frameworks, applied to the data used in the SAMURAI project [sam]. SAMURAI developed robust moving object, segmentation, categorisation and tagging in video captured by multiple cameras from medium-long range distance, e.g. identifying, monitoring and tracking people with luggage between different locations at an airport. Automated focus of attention and identification in a distributed sensor network that includes fixed and mobile cameras, positioning sensors, and wearable audio/video sensors. Global situational awareness assessment and image retrieval of objects by types, movement patterns with incidents across a distributed network of cameras. Online adaptive abnormal behaviour monitoring for profiling and inference of abnormal behaviours/events captured by multiple cameras. One of the goals of this project is to reveal the presence of a pedestrian by drawing a bounding box surrounding each person. The detected people are then labelled according to their appearance, posture, kinematics, and their association with luggage and vehicles.

At the beginning of the SAMURAI project the adopted pedestrian detection was the one built upon the binary LogitBoost method on Riemannian Manifolds as described in Sec. 5.2, that it is renamed as FUD. This solution provides good results for medium/high resolution pedestrians, but performs poorly with low resolution pedestrians. To improve it, its training and testing procedures are significantly modified adopting a regular grid set of patches to characterize a pedestrian as described in Sec. 5.6, which is denoted by EOD. This solution permits to enhance the robustness of the pedestrian detector, being intrinsically more robust to occlusions, noise, and able to deal with very low resolution pedestrians. One may ask why the other pedestrian detection frameworks presented in this Chapter do not appear in this experimental section: it is due to their computational burden. FAD and EOD are used because they can exploit the integral representation [TPM06] that leads to a light-speed computation of the descriptors.

The comparison is made in two different test sites. The test site depicted in Fig. 5.25 has been set up in one of the project partner company, Elsag Datamat S.P.A., in Italy. It allows to facilitate validation of the SAMURAI architecture. This site contains a car park. The goal is to ensure visitors register themselves at the security check-in counter/reception building before they enter the office building. The behaviour of not registering themselves is considered abnormal. In this scenario the cameras cover different views of the car park.

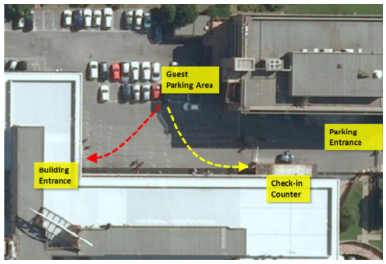


Fig. 5.25. Elsag Datamat test site.

The third terminal of the Heathrow Airport in the United Kingdom, depicted in Fig. 5.26, is the other test site. In particular, where passengers usually drive into the area, park their cars and get into the airport departure or arrival areas. The goal is to detect the abandonment of unauthorised vehicles (except coaches, police cars, emergency vehicles and taxi) in the inner lane of the forecourt. Timely detection of this situation is critical for enhancing the security at the airport. In this scenario the cameras cover different areas of the forecourt, i.e. the inner lane of the forecourt, the car park, and the lobby at the terminal building.

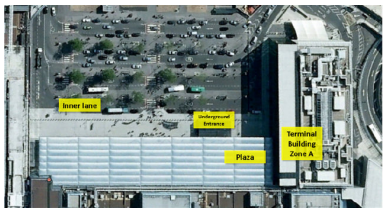


Fig. 5.26. T3 Heathrow Airport test site.

The qualitative results shown in Sec. 5.7.1 for the low resolution pedestrian detection task and in Sec. 5.7.2 medium/high resolution pedestrian detection task are a little part of the detection results obtained for the SAMURAI project. Unfortunately, a ground truth is not available for the SAMURAI data, however in this Section the different behaviour given by the FUD and the EOD detectors is highlighted in real settings, since the quantitative statistics not always tell the truth about the effectiveness of a detector as demonstrated in [DWSP11].

Before showing and commenting the results, it is worth noting that: (1) both the detectors (FUD and EOD) are trained using the INRIA Person Dataset [Dal05], so they do not contain images coming from the test sites; (2) the results shown are not post-processed by any non-maximum suppression method in order to show the actual output of the detectors.

5.7.1 Low Resolution Pedestrian Detection

For what concerns the low resolution pedestrian detection task, the superior performance of the EOD detector in comparison with the FUD detector is shown.

This because the FUD framework as [TPM08], which is the most similar detector, relies on many small details of a set of medium size pedestrians. So, at low resolution these small details become unreliable and the detector fails. Here the big region of the regular grid on which EOD is built and the compactness of the covariance descriptor lead to a very effective detector in the low resolution case.

(a)

(b)

Fig. 5.27. Low resolution pedestrian detection examples. (a) The results given by the EOD framework described in Sec. 5.6. (b) The results given by the FUD framework described in Sec. 5.2

5.7.2 Medium/High Resolution Pedestrian Detection

On the contrary, when pedestrian are medium or high, a resolution FUD classifier can use the small details to discriminate better the presence of a pedestrian in an image. In this case, a EOD detector perform poorly because of the compactness of the few covariance matrices on which it is based.

(a)

(b)

Fig. 5.28. Medium/high resolution pedestrian detection examples. (a) The results given by the EOD framework described in Sec. 5.6. (b) The results given by the FUD framework described in Sec. 5.2

6

Classification using Tensors

Contents

6.1 Introduction

In video surveillance a pedestrian, detectors like those presented in Chap. 5, can be seen as the first step towards a people tagging or re-identification module. Also

in this case, the issue of how to represent a person for those high level tasks is central. Recalling the good results obtained by tensors in Chap. 4 for multi-class classification tasks, some of those tensors are exploited for people categorization tasks. The main contribution of this Chapter are: a new class of features referred to as ARCO which is further evolved to WARCO and FWARCO for the description of low resolution objects on different regression and multi-class problems, such as head orientation classification, human orientation classification, pedestrian classification, head pose estimation. For all these tasks novel datasets are introduced. In addition, it introduces a novel criterion, based on the Riemannian curvature, to estimate the non-flattens of a set of tensors, which can be used to estimate the error committed in approximating tensors on a Euclidean manifold for learning purposes. That criterion is valid over any connected Riemannian manifold. Besides, it describes a way to find possible approximations of the actual distance among tensors that can be combined with standard machine learning algorithms for multi-class classification and regression problems.

As for the previous Chapter, the attention is focused on people in the video surveillance context. As a first attempt of tagging a person, the problem of categorizing his head orientation is tackled. This is because the orientation of the head allows to infer which part of the scene is observed by that person, which is useful to infer what a person may be interested in, or to understand whether a person is interacting with another one. So, in Sec. 6.2 the binary LogitBoost on Riemannian Manifolds (see Sec. 5.2) is extended to the multi-class case, employing it to detect head orientations. Even if the results obtained are promising, the computational cost of the learning phase of this solution is prohibitive for large datasets. This prompts me to find a different learning strategy able to manage efficiently SPD tensors for multi-class problems. As results, in Sec. 6.3, a novel feature, the ARray of COvariances (ARCO), is proposed, and a multi-class classification framework operating on Riemannian manifolds is introduced. ARCO is composed of a structure of covariance matrices of image features, able to extract information from data at prohibitive low resolutions. The proposed classification framework consists in instantiating a new multi-class boosting method, working on the manifold Sym_d^+ of symmetric positive definite $d \times d$ (covariance) matrices. As practical applications, different surveillance tasks are considered, such as head pose classification and pedestrian detection, providing novel state-of-the-art performances on standard datasets.

In Sec. 6.4 Weighted ARCO (WARCO) is presented: it represents a significant revision and extension of ARCO. It revisits this feature reporting a comprehensive theoretical analysis that motivates some fundamental choices with regard how it is possible to compute the distance among covariance matrices. Moreover, the study goes a step further proposing different approximations of that distance and showing the goodness of this framework in both theoretical and empirical ways. Moreover, with WARCO, a more effective and efficient statistical framework is introduced, if compared to the one proposed in Sec. 6.4. A thorough evaluation is finally provided, on several public datasets, specifically devoted to head orientation classification, human body pose classification, and head orientation estimation in real surveillance scenarios, showing that the proposed method outperforms in most of the cases the state-of-the-art results.

Then in Sec. 6.5, Fast WARCO (FWARCO) is introduced for fast and robust inference, which exploits Random Forest (described in Sec. 3.3.2) to train the WARCO patch models. This because RF is very efficient both in the training and testing phases. Moreover, in terms of robustness, a hard negative mining strategy is designed for RF, that is particularly useful for multi-class classification problems where the background class is present.

Exploiting ARCO, in Sec. 6.6 a multi-class detection framework is built to model the Visual Focus of Attention (VFOA) of a person which is a very important cue in human behaviour analysis. VFOA classification is difficult, though, especially in an unconstrained and crowded environment, typical of video surveillance scenarios. In this Section, VFOA is estimated by defining the Subjective View Frustum, which approximates the visual field of a person in a 3D representation of the scene. This opens up to several intriguing behavioural investigations, in particular the proposed *Inter-Relation Pattern Matrix*, which suggests possible social interactions between the people present in a scene.

Sec. 6.7 tries to understand if it is possible to exploit the tensor representation to build a more powerful object descriptor compared to COV (Covariance) representation for several multi-class classification problems. To that end, in Sec. 4.2.2 EMI (Entropy-Mutual Information) tensor is introduced and Sec. 4.3 shows that EMI beats COV tensor. This experimental session completes the one started in Sec. 4.3, providing to EMI and COV a more complex object model in the same spirit of [BZM07a], thus using a spatial pyramid representation.

6.2 Multi-class LogitBoost on Riemannian Manifolds: A Direct Extension

Tuzel et al. [TPM08] have recently concentrated on the use of covariance features as human descriptors. A region is represented by the covariance matrix of image features, such as spatial location, intensity, gradient values etc. Within the context of human detection, these matrices are associated with different overlapping subregions inside the detection window containing the whole human body. Since covariance matrices lie in the Riemannian manifold $\mathcal{M}$ of the SPD matrices denoted by Sym^+, a modified version, working on Riemannian manifolds, is proposed in Tuzel et al. [TPM08]. This Section examines Tuzel et al.'s approach and proposes the extension of their method to the multi-class classification case. The obtained framework is then employed for head orientation classification.

6.2.1 Learning Framework

Assume to have a J-class classification problem. Let $\mathcal{S} = \{\mathbf{X}_i, y_i\}_{i=1,\dots,N}$ be the set of training examples, with $y_i \in \{1, \dots, J\}$ and $\mathbf{X}_i \in \mathcal{M}$. The goal is to find a set $F = \{F_j\}_{j=1,\dots,J}$ of response functions, for $F(\mathbf{X}_i) : \mathcal{M} \mapsto \{1, \dots, J\}$, that divides the input space into J parts, based on the training set of labelled items. F_j is a single class strong classifier and is defined as a sum of weak classifiers. According to [TPM08], an incremental approach is adopted by training locally different sets of weak learners on tangent spaces of $\mathcal{M}$ and then combining them

with the boosting model. The essence of a boosting algorithm is an iterative re-weighting system that tends to focus on the most difficult examples in the training set. In the multi-class case, J different sets of weights can be built by the posterior distributions. Let $\mathrm{P}(j|\mathbf{X}_i)$ $(= \mathrm{P}_j(\mathbf{X}_i)$) be the posterior probability for $\mathbf{X}_i$ being in the j-th class. It is represented by:

$$\underset{j}{\mathrm{P}}(\mathbf{X}_i) = \frac{e^{F_j(\mathbf{X}_i)}}{\sum_{k=1}^{J} e^{F_k(\mathbf{X}_i)}}, \quad F_j(\mathbf{X}_i) = \sum_{l=1}^{L} f_{lj}(\mathbf{X}_i), \tag{6.1}$$

where $\{f_{lj}\}_{l=1,\ldots,L}$ is a class-specific set of weak learners. Every example in $\mathcal{S}$ is associated with a weight that depends on the class considered. Considering $\mathbf{X}_i \in \mathcal{S}$ and the j-th class, its weight can be calculated as:

$$w_{ij} = \underset{j}{\mathrm{P}}(\mathbf{X}_i)(1 - \underset{j}{\mathrm{P}}(\mathbf{X}_i)). \tag{6.2}$$

At the core of the learning process, the decision boundaries are built by the weak learners. These are simple lines fitted by solving a weighted lest-square regression problem. To measure the goodness of the regressors one can use the response values z. As for the weights, response values are class-specific and are defined for every example in $\mathcal{S}$ as:

$$\mathbf{z}_{ij} = \frac{y^*_{ij} - \mathrm{P}_j(\mathbf{X}_i)}{\mathrm{P}_j(\mathbf{X}_i)(1 - \mathrm{P}_j(\mathbf{X}_i))}, \tag{6.3}$$

where $\mathbf{Y}^*_{ij} = 1j = y_i$, with $1\{\cdot\}$ an indicator function that takes 1 if and only if $(j = y_i)$ is true, and $\mathbf{y}^*_i = [\mathbf{Y}^*_{i1}, \ldots, \mathbf{Y}^*_{iJ}]$. The main difference between multi-

Algorithm 10: Multi-class LogitBoost on $\mathcal{M}$

Data: $(\mathbf{X}_1, y_1), \ldots, (\mathbf{X}_N, y_N)$ with $\mathbf{X}_i \in \mathcal{M}$ and $y_i \in \{1, \ldots, J\}$.
Result: The multi-class classifier F.
begin
  $\forall i$ $\forall j$ start with weights $w_{ij} = 1/N$ and $i = 1, \ldots, N$, $F_j(\mathbf{X_i}) = 0$ and $\mathrm{P}_j(\mathbf{X}_i) = 1/J$;
  for $l = 1, 2, \ldots, L$ *(L = total number of weak learners)* **do**
    for $j = 1, 2, \ldots, J$ **do**
      Compute the response values and weights with Eq. (6.3) and Eq. (6.2) respectively;
      Compute the weighted mean $\boldsymbol{\mu}_{lj}$ of the j-th class points through (5.4);
      Map the data points to the tangent space at $\boldsymbol{\mu}_{lj}$ and then vectorize them as in (2.1). $\mathbf{x}_i \in \mathbb{R}^n$ denotes a point on the tangent space;
      Fit the binary function g_{lj} by weighted least-square regression of $\mathbf{z}_{ij}$ to $\mathbf{x}_i$ using weights w_{ij};
      Set $F_j(\mathbf{X}) \leftarrow F_j(\mathbf{X}) + f_{lj}$ where f_{lj} is defined in Eq. (6.2.1) and G_{lj} from Eq. (6.5);
      Update $\mathrm{P}_j(\mathbf{X})$ as in Eq. (6.1);
      $F_j = F_j \cup \{\mu_{lj}, g_{lj}\}$;

class LogitBoost on $\mathcal{M}$ and $\mathbb{R}^n$ is at the weak learners level. In the $\mathcal{M}$ case a weak learner is defined as a map $f_l(\mathbf{X}) : \mathcal{M} \mapsto \mathbb{R}$, whereas in the $\mathbb{R}^n$ case $f_l(\mathbf{x}) : \mathbb{R}^n \mapsto \mathbb{R}$.

The weak learners are defined as:

$$f_l(\mathbf{X}) = g_l(\underset{\boldsymbol{\mu}_l}{\text{vec}}(\log_{\boldsymbol{\mu}_l}(\mathbf{X}))), \tag{6.4}$$

and the regression functions $g_l(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ are learned on the tangent space $T_{\boldsymbol{\mu}_l}\mathcal{M}$ at the weighted mean $\boldsymbol{\mu}_l \in \mathcal{M}$ of the points, estimated as in Eq. (5.4). Notice that the mapping vec (defined in Sec. 2.2.6) gives the orthonormal coordinates of the tangent vectors in $T_{\boldsymbol{\mu}_l}\mathcal{M}$.

The multi-class extension of the basic binary learner is instead the same for $\mathcal{M}$ and $\mathbb{R}^n$. This is because the extension is done after projecting all the training data in an appropriate class-dependent tangent space. A *multi-class weak learner* is defined as in [ZPG+06]:

$$G_{lj}(\mathbf{x}_i) = \frac{J-1}{J}\left(g_{lj}(\mathbf{x}_i) - \frac{1}{J}\sum_{k=1}^{J} g_{lk}(\mathbf{x}_i)\right), \tag{6.5}$$

where $g_{lj}(\mathbf{x}_i)$ is a binary classifier for class j and $\mathbf{x}_i$ is an element of $\mathcal{M}$ mapped on the tangent space of $\boldsymbol{\mu}_l$. G_{lj} represents a sort of disparity from the mean of all the binary weak classifiers responses.

A description of multi-class LogitBoost working on $\mathcal{M}$ is illustrated in Alg. 10.

6.2.1.1 Class-specific dense object model

Recalling that the goal is to represent an object as a set of covariance matrices, this type of object representation is known as *dense* object model. In order to select, at each iteration of the learning process, the best features (i.e. the best sub-window inside the window box where to estimate the covariance matrices), a two steps greedy optimization approach is proposed. The underlying idea is, first, to optimize features selection at class level. Then, the best features combination among all classes is optimized globally utilizing the results from the first step. This approach is followed in order to maximize efficiency.

Assuming that each sub-window is one-to-one with a weak learner, a *shared set of weak learners* $\{f_t\}_{t=1,\dots,T}$ is selected. T is arbitrarily chosen, typically 200 in the experiments. According to Eq. (6.1), the posterior for every weak learner is computed. Then, the weak learners are ordered for each class separately, according to the binomial log likelihood $l_{tj}(\cdot,\cdot)$:

$$l_{tj}(\mathbf{y}^*, \mathrm{P}(\mathbf{X})) = \sum_{i=1}^{N} y^*_{ij} \log \underset{jt}{\mathrm{P}}(\mathbf{X}_i) + (1 - \mathbf{Y}^*_{ij}) \log(1 - \underset{jt}{\mathrm{P}}(\mathbf{X}_i))\}, \tag{6.6}$$

where P_{jt} is the posterior for class j on a sub-window t.

In the second step, the best combination of sub-windows tuple

$$T_m = [t_{1m}, t_{2m}, \dots, t_{Jm}]$$

is selected. The candidates are the tuples derived from the sorting in the first step, such that $T_k = [t_{1k}, t_{2k}, \dots, t_{Jk}]$ is composed by the J sub-windows at the k-th

Algorithm 11: Multi-class LogitBoost on $\mathcal{M}$ with dense object model

Data: $(\mathbf{X}_1, y_1), \ldots, (\mathbf{X}_N, y_N)$ with $\mathbf{X}_i \in \mathcal{S}$ and $y_i \in \{1, \ldots, J\}$.
Result: The multi-class classifier F.
begin
  $\forall j$ start with weights $w_{ij} = 1/N$ e $i = 1, \ldots, N$, $F_j(\mathbf{X}) = 0$ and $\mathrm{P}_j(\mathbf{X}_i) = 1/J$;
  for $l = 1, 2, \ldots, L$ **do**
    Sample T sub-windows and construct covariance matrices;
    for $j = 1, 2, \ldots, J$ **do**
      for $t = 1, 2, \ldots, T$ **do**
        Compute the response values and weights with Eq. (6.3) and Eq. (6.2) respectively;
        Compute the weighted mean μ_{ijt} of the j-th class points through (5.4);
        Map the data points to the tangent space at μ_{ijt} and then vectorize them as in (2.1). $\mathbf{x}_i \in \mathbb{R}^n$ denotes a point on the tangent space;
        Fit the function g_{ljt} by weighted least-square regression of z_{ijt} to $\mathbf{x}_i$ using weights w_{ijt};
    Select the best ensemble of weak classifiers $\{f_{lj}\}_{j=1,\ldots,J}$, with the two step optimization;
    Order the covariance descriptors with Eq. (6.6) $\forall j$;
    Select the best combination with Eq. (6.7);
    for $j = 1, 2, \ldots, J$ **do**
      Set $F_j(\mathbf{X}) \leftarrow F_j(\mathbf{X}) + f_{lj}$ where f_{lj} is defined in Eq. (6.2.1) and G_{lj} as in Eq. (6.5);
      Update $\mathrm{P}_j(\mathbf{X})$ as in Eq. (6.1);
      $F_j = F_j \cup \{T_m, \boldsymbol{\mu}_{lj}, g_{lj}\}$;

position in the classes $1, 2, \ldots, J$, respectively. T_m is estimated by minimizing the negative multinomial log-likelihood of the data:

$$L_k(\mathbf{y}^*, \mathrm{P}(\mathbf{X})) = -\sum_{i=1}^{N}\sum_{j=1}^{J} y_{ij}^* \log \underset{j,t_{jk}}{\mathrm{P}}(\mathbf{X}_i) + (1 - \mathbf{Y}_{ij}^*) \log(1 - \underset{j,t_{jk}}{\mathrm{P}}(\mathbf{X}_i))\}. \quad (6.7)$$

Here, $\mathrm{P}_{j,t_{jk}}$ is the posterior estimated for class j and sub-window t_{jk}.

To show the effect of using this strategy on the multi-class LogitBoost, Alg. 11 gives a detailed description.

6.2.1.2 Tree structured classification

Different classes have different intrinsic complexity. The more a class is compact in the features space, the easier its learning process is. It means that each class needs its own number of weak learners to be learned. Therefore, it is necessary to establish a stop criteria to define when a class is learned. When all classes are learned, the learning process terminates. At the state-of-the-art, only in [ZPG+06] a stop criterion is proposed. This criterion is defined for detection purposes, and it

is directly based on the strong classifier response $F(\mathbf{X})$. Since F takes values in $\mathbb{R}$, their criteria has only an empirical interpretation. So, a new criteria based on posterior probability $\mathrm{P}(\mathbf{X})$ is established, with a clearer probabilistic interpretation. The proposed condition is defined by two parts, one regarding the posterior probability of the elements belonging to a class, and the other concerning the elements that do not belong to the same class. The learning of the j-th class is stopped when at least $TH\%$ of the examples in that class, C_j, are correctly classified, that is to say when the following condition holds:

$$\underset{j}{\mathrm{P}}(\mathbf{X}_i) > \underset{k}{\mathrm{P}}(\mathbf{X}_i),\ \mathbf{X}_i \in C_j,\ k \neq j \in \{1, \ldots, J\}. \tag{6.8}$$

Moreover, one may want that most of the other examples belonging to the other classes are not wrongly classified as belonging to C_j:

$$\underset{j}{\mathrm{P}}(\mathbf{X}_i) < \underset{k}{\mathrm{P}}(\mathbf{X}_i),\ \mathbf{X}_i \in C_k,\ k \neq j. \tag{6.9}$$

Also in this case one may want that (6.9) holds for at least $TH\%$ of the examples in C_k. It is possible to make the conditions in Eq. (6.8) and (6.9) more restrictive by adding a probabilistic margin $marg$, i.e:

$$\underset{j}{\mathrm{P}}(\mathbf{X}_i) + marg > \underset{k}{\mathrm{P}}(\mathbf{X}_i),\ \mathbf{X}_i \in C_j,\ k \neq j \in \{1, \ldots, J\}, \tag{6.10}$$

and

$$\underset{j}{\mathrm{P}}(\mathbf{X}_i) < \underset{k}{\mathrm{P}}(\mathbf{X}_i) + marg,\ \mathbf{X}_i \in C_k,\ k \neq j. \tag{6.11}$$

Assuming that classes have a different number of weak learners, it is necessary to define how to compute the posterior probability. As in [ZPG+06], a *tree structure* is adopted. A toy problem with 3 classes ($J = 3$) is used to explain the posterior computation. Suppose that after several LogitBoost iterations the 1-st class satisfies the stop criterion. The learning for this class is stopped, thus creating the first layer of the tree. The posterior probability $p_{1,1}(\mathbf{X})$ of the 1-st learned class at the first tree layer (the former sub-index in p indicates the class, the latter the three layer) is computed, according to Eq. (6.1), as:

$$\underset{1,1}{\mathrm{P}}(\mathbf{X}) = \frac{e^{F_{11}(\mathbf{X})}}{\sum_{k=1}^{J} e^{F_{1k}(\mathbf{X})}}. \tag{6.12}$$

The sub-indexes in F has the same meaning as in p. At the next tree layer one should remove the 1-st class and all the remaining classes share the residual posterior probability:

$$R_{\{2,3\},1}(\mathbf{X}) = 1 - \underset{1,1}{\mathrm{P}}(\mathbf{X}). \tag{6.13}$$

Now, let us suppose that the second class is learned at the second layer of the tree. Its posterior probability becomes:

$$p_{2,2}(\mathbf{X}) = \frac{e^{F_{22}(\mathbf{X})}}{\sum_{k=2}^{J} e^{F_{2k}(\mathbf{X})}} \cdot R_{\{2,3\},1}(\mathbf{X}), \tag{6.14}$$

and the residual of this second layer is

$$R_{3,2}(\mathbf{X}) = (1 - \underset{2,2}{\mathrm{P}}(\mathbf{X})) \cdot R_{\{2,3\},1}(\mathbf{X}). \tag{6.15}$$

Therefore, in general the posterior probability for the j-class at the u-th tree layer is built as

$$p_{j,u}(\mathbf{X}) = \frac{e^{F_{jl}(\mathbf{X})}}{\sum_{k \in J_u} e^{F_{lk}(\mathbf{X})}} \cdot \prod_{i=1}^{l-1} R_{ji}(\mathbf{X}), \tag{6.16}$$

where $\prod_{i=1}^{l-1} R_{ji}(\mathbf{X})$ is the product of the $(l-1)$ previous residues, and J_u is the set of classes active at level u.

6.2.2 Experimental Results

6.2.2.1 Human Detection

Before to test the proposed framework on a multi-class classification problem, a binary problem is considered, i.e. the pedestrian detection. The human detector is trained on INRIA Person dataset [DT05]. In the first experiment, adopting the improvements described in Sec. 5.2, a toy example is shown, training a cascade of 11 levels with 500 positive examples and 1000 negative examples per level, in order to show the behaviour of the framework in the training phase. The proposed method is compared with a standard random selection of negative examples. In Fig. 6.1, the x-axis and the y-axis correspond to the cascade level and to the number of weak learners per level respectively. Considering the number of weak learners of the experiment, a cut of the 45% in the cascade complexity is obtained, evaluated as number of classifiers, in comparison to the normal cascade. Both classifiers obtain similar detection performances, but the proposed method is more efficient, because of the minor number of classifiers to be evaluated.

Experiments are conducted on two challenging real video surveillance scenarios. The first one is a home-made video that portrays an indoor coffee-room scene [Baz], where students take coffee or discuss. This resembles a restricted surveillance area, like a shop or a bar inside an airport. The video footage is acquired with an off-the-shelf monocular camera, located on a upper angle of the room. This video is particularly challenging due to shadows and reflections (on the floor and on the coffee machines). The second one is from a publicly available dataset, PETS2007 [pet], and depicts a scene inside an airport. Even in this case the lighting conditions are quite challenging, with a bright sun entering trough the windows.

The classifier is trained using a 22-levels cascade. A total of 3000 positive examples are extracted, joining PETS2007 and INRIA Person datasets [Dal05], and 5400 negative examples per level are employed. All the examples are scaled into a fixed size of 64×128, which includes a small margin all around the pedestrians. The negative examples come from 1218 person-free images of the INRIA person data set and about 100 person-free patches of PETS2007. In Fig. 6.2 plots the number of weak classifiers at each cascade level and the accumulated rejection rate over the cascade levels. A curve's inflection it can be noticed at level 4. This is due to the transition from the training with only *easy* negative examples to the training with the *hard* ones. The performance of the proposed classifier on INRIA person data test set are measured. The Detection Error Tradeoff (DET) curve is

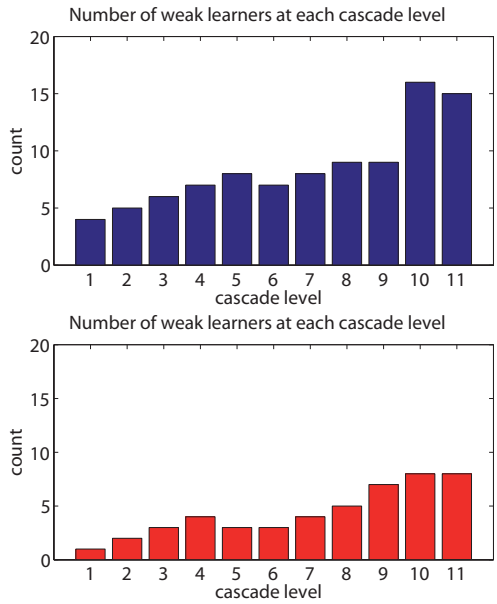


Fig. 6.1. The cascade of classifiers produced in training phase for a toy example. On the top, a random sampling methodology is adopted; on the bottom, the proposed methodology based on the low level semantic.

adopted. The curve is depicted in Fig. 6.2. Compared to the results in [TPM08] similar performances are obtained, even to with a less number of weak learners and cascade levels.

Detection is carried out by checking each window inside the detection image. Different scales (0.5, 0.7, 1.0, 1.5) are checked by scaling the image accordingly and applying the classifier at the original scale. The positive detections are filtered out by selecting the local maxima of the detection outputs.

Some detection results are shown in Fig. 6.3 for the coffee-room sequence and in Fig. 6.4 for S08 sequence in PETS 2007. As to the coffee-room sequence, 3580/5955 people are detected, about 60%, with only 53 false positives on a total of 2408 frames of dimension $[600 \times 384]$. The false negatives are mainly due to occlusions, since the room is very small and people tend to gather together, forming groups. The number of false positive is very small, though. As to the PETS 2007 sequence, the proposed method detects 1310/2050 people, about 64%, and 320 false positives on 660 frames of dimension $[720 \times 576]$. In this case, the false negatives are due to occlusions and the lighting conditions. The false positives, instead, are mainly concentrated on the back of the hall, at the border between illuminated and shadowed area.

In these tests human detection at each frame is performed. The percentage of false negatives can certainly be increased by integrating the background subtraction module and by adding spatio temporal considerations.

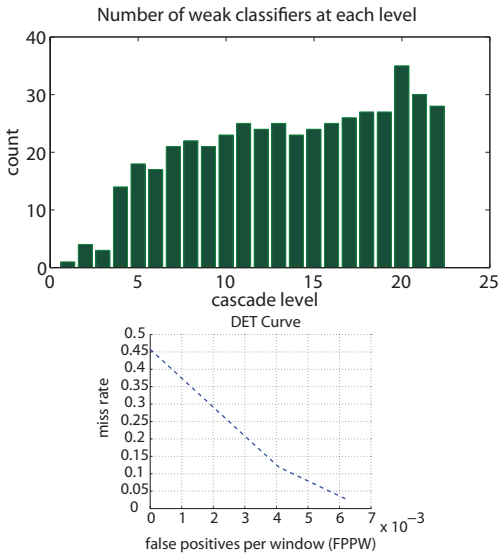


Fig. 6.2. The number of weak classifiers at each cascade level on the top. On the bottom, the DET curve on the INRIA test set.

Fig. 6.3. Detection examples on the coffee-room sequence [Baz]. Red dots are the local maxima of detection outputs, and the rectangles are the averaged detection window sizes.

Fig. 6.4. Detection examples on PETS2007 [pet] S08 sequence.

6.2.2.2 Head Pose Orientation

The multi-class framework on Riemannian manifolds is applied on the classification of head pose orientation. Since I want to work in realistic video surveillance scenarios, where usually pedestrians are rather small in the image, so a quite rough estimation of head orientation is enough. For video surveillance it is sufficient to classify 4 orientations: front, back, left and right. A training set composed by 46×60 head examples is built, taken from a subsequence of the coffee-room sequence, with about 200 examples for each class. All positive examples contain a margin of about 10 pixels. This makes the classifier much more robust. I trained the classifier on 5 classes (one is the background) with the tree structure, until the stop criterion holds for each class. 20000 negative examples from the INRIA dataset are used.

The classifier has been tested on the rest of the coffee-room sequence. The one-third upper part of a pedestrian positive detection window is selected, as detected by the human detector above, and the classifier is applied. The detection rates are shown in Tab. 6.1. Some correctly classified head images are illustrated in Fig. 6.5.

TH%	Back	Front	Left	Right
80	55	64	68	67
85	75	77	78	70
90	85	82	91	79

Table 6.1. Classification performance (in percentage) with different training sets at different detection thresholds (TH).

Fig. 6.5. Some examples of correctly classified images in [Tosb], frontal, left, right and backward orientation respectively.

6.3 ARCO (ARray of COvariance Matrices)

Even if the results obtained in the previous Section are promising, the computational cost of the learning phase of that solution is prohibitive for large datasets. This prompts me to find a different learning strategy able to manage efficiently SPD tensors for multi-class problems. Therefore In this Section, a novel feature is proposed, the ARray of COvariances (ARCO), and an efficient multi-class classification framework operating on Riemannian manifolds for video surveillance purposes.

An important goal of automated video surveillance is to design algorithms that can characterize different objects of interest (OIs), especially if immersed in a cluttered background and captured at low resolution. The detection (e.g. of faces or pedestrians) and the classification (e.g. of facial poses) are among the most studied applications. In the multi-faceted plethora of approaches in the literature (see [MCT09, EG09, YKA02] for extensive reviews), boosting-based techniques play a primary role [VJ01, LZZ+02, VJV03, HALL05, WAHL04, LZ04, BHHW05, TPM08, YO08, WN08, PSZ08]: boosting [FS97, SS99, FHT00] is a remarkable, highly customizable way to create strong and fast classifiers, employing various features fed into diverse architectures, with specific policies.

Among the different features considered for boosting in object classification (see [WN09] for an updated list), covariance features [TPM06] have been exploited as powerful descriptors of pedestrians [TPM08, YO08, WN08], and their effectiveness has been explicitly investigated in a comparative study [PSZ08]. When injected in boosting systems [TPM08, YO08, WN08, PSZ08], covariances provide strong detection performances. They encapsulate the high intra-class variances (due to pose and view changes of the OI), they are in general stable in presence of noise, and provide an elegant way to fuse multiple low-level features, as they intrinsically exploit possible inter-features' dependencies. Moreover, thanks to the integral image representation, they can be calculated in a very efficient way.

Since covariance matrices lie in the Riemannian manifold of symmetric positive definite matrices Sym_d^+, their usage in a boosting framework requires a careful treatment. In [TPM08], the input covariance features are projected into the tangent space at particular points of the manifold, where an Euclidean metric can be instantiated, and the Logitboost framework can be applied.

In this Section, two main contributions are proposed. First, a novel kind of feature is presented, *i.e.* the *ARray of COvariances* (ARCO), able to describe visual objects at prohibitive low resolutions (up to 5×5 pixels): it marries the dense descriptors philosophy, adopted for example in [DT05], with the expressivity of the covariance information. Second, it is shown how such features can be embedded in a multi-classification framework by boosting, extending [TPM08] to the multi-class case. It turns out that Sym_d^+ has non positive curvature and in the areas where the curvature is almost flat the Euclidean metric on the tangent space at any point on the manifold is a good approximation of the Riemannian metric. Therefore, unlike [TPM08], all the data is mapped in a unique tangent space, and all the computations are performed on this (Euclidean) space where a typical multi-class LogitBoost (see Alg. 3) algorithm can be applied.

The experimental trials show how the proposed method outperforms in two important applications for surveillance like head pose classification and pedestrian detection, without adopting complex boosting schemes such as Floatboosting for pyramids [LZ04], decision trees [VJV03], VectorBoosting for width-first-search trees [HALL05], or Probabilistic Boosting Networks [ZZMC07]. Novel state-of-the-art performances on standard databases are fixed. This encourages the embedding of the proposed Riemannian framework in the above quoted boosting schemes. I stress also the capability of dealing with compelling image resolutions, promoting the use of ARCOs for heterogeneous applications, especially in the surveillance field.

The rest of the Section is organized as follows. Sec. 6.3.1 describes the proposed ARCO feature and Sec. 6.3.2 depicts the proposed multi-class framework. Sec. 6.3.3 shows the experimental results on several surveillance applications.

6.3.1 ARCO: ARray of COvariance Matrices

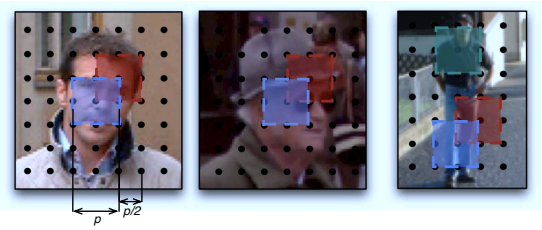


Fig. 6.6. Array of Covariance matrices (ARCO) feature. The image is organized as a grid of uniformly spaced and overlapping patches. On each patch, a multi-class classifier is estimated.

The proposed classification framework has been specifically designed to deal with low resolution images, typical of a video surveillance scenario. In such conditions, the number of features that can be extracted is relatively small, and quite unreliable. This is very challenging in problems like, for instance, head pose classification, in which the details are crucial to distinguish the different object classes. Moreover, the classifier must cope with object (pedestrians, heads) views in a variety of light conditions. The solution is based on two main concepts: 1) the organization of the image into a grid of uniformly spaced and overlapping patches (Fig. 6.6); 2) the use of covariance matrices of image features as patch descriptors, which are classified by multi-class LogitBoost on Riemannian manifolds. To summarize, each patch classifier votes for a class, and the final classification result is the class voted by the majority of them.

In [TPM08], where the use of covariance matrix descriptors is tailored for pedestrian detection, LogitBoost was used both for a greedy estimation of the most discriminative patches among a set of different sizes and positions, and for classifying them, i.e., as feature selection and classification method at the same time. The same reasoning, using boosting for feature selection and classification, has been applied to other approaches in the literature, as for example in [WN09, YTCC09]. Here, instead, a feature selection operation is infeasible, because low resolution images contain such scarce and noisy information that the result would be unreliable: it is more convenient to use *all* features in a suitable way. The proposed approach draws inspiration from the literature on dense image descriptors (see [DT05] for example). The image I is sampled into uniformly distributed and overlapping patches of the same dimension. Each patch is described by the covariance matrix representation, that encodes the local shape and appearance of the (small) region. These patches are used in a democratic way: exalting their discriminative

power by boosting a strong multi-class classifier, and collecting their classification results.

More formally, given a set of patches $\{P_i\}_{i=1,\ldots,N_P}$, a multi-class classifier is learned for each patch $\{F_{P_i}\}_{i=1,\ldots,N_P}$ through the multi-class LogitBoost algorithm [FHT00], adapted to work on Riemannian manifolds.
Let $\Delta_j = \sum_{i=1}^{N_P}(F_{P_i} == j)$ be the number of patches that vote for the class $j \in \{1,\ldots,J\}$. A class label c is assigned to an image, estimating

$$c = \arg\max_j \{\Delta_j\}, \quad j = 1,\ldots,J. \tag{6.17}$$

In order to increase robustness to local illumination variations, the normalization operator introduced in [TPM08] is utilized before applying the multi-class framework.

The ARCO representation has several advantages. First, it allows to take into account different features, inheriting their expressivity, and exploiting possible correlations. In this sense, it is as a compact and powerful integration of features. Second, due to the use of integral images, ARCO is fast to compute, making it suitable for a possible real-time usage. Finally, as a dense representation, it is robust to occlusions. All the characteristics above will proved during the experimental trials in Sec. 6.3.3.

6.3.2 Multi-class Classification on Riemannian Manifolds

Let $C_1, C_2, \ldots, C_J$ be the data classes whose elements (the covariances) live in the Riemannian manifold $\mathcal{M}$ of $d \times d$ symmetric positive definite matrices denoted by Sym_d^+. Let $\mathcal{S} = \{\mathbf{X}_i, y_i\}_{i=1,\ldots,N}$ be the set of N training examples, with $\mathbf{X}_i \in \mathcal{M}$ and label $y_i \in \{1,\ldots,J\}$. The aim is to produce a function $F(\mathbf{X}_i) : \mathcal{M} \mapsto \{1,\ldots,J\}$ as

$$F(\mathbf{X}_i) = \arg\max_j \{F_j(\mathbf{X}_i)\}, \quad j = 1,\ldots,J. \tag{6.18}$$

F_j is a *single-class* strong classifier, and it is defined, in turn, as a sum of L weak classifiers $\{f_{lj}\}_{l=1,\ldots,L}$. These weak classifiers are learned by multi-class LogitBoost.

6.3.2.1 Riemannian Geometry on Sym_d^+

In this section, the geometry of Sym_d^+ is briefly reviewed, since the manifold consists of all $d \times d$ symmetric definite positive matrices (covariance matrices), extending the treatment given in [TPM08].

The tangent space $T_{\mathbf{Y}}$ at any point $\mathbf{Y} \in Sym_d^+$ can be identified with Sym_d, the (vector) space of $d \times d$ symmetric matrices.

The mapping of $\mathbf{X}$ on $T_{\mathbf{Y}}$ is given by the point-dependent $\log_{\mathbf{Y}}$ operator:

$$\log_{\mathbf{Y}}(\mathbf{X}) = \mathbf{Y}^{\frac{1}{2}} \log\left(\mathbf{Y}^{-\frac{1}{2}}\mathbf{X}\mathbf{Y}^{-\frac{1}{2}}\right)\mathbf{Y}^{\frac{1}{2}}, \tag{6.19}$$

inverse to the exponential map.

The (geodesic) distance on Sym_d^+ is defined as

$$d^2(\mathbf{X}_1, \mathbf{X}_2) = \mathrm{tr}(\log(\mathbf{X}_1^{-\frac{1}{2}} \mathbf{X}_2 \mathbf{X}_1^{-\frac{1}{2}})^2) = \sum_{i=1}^{d} (\log \xi_i)^2 \tag{6.20}$$

where the ξ_i's are the (positive) eigenvalues of $\mathbf{X}_1^{-\frac{1}{2}} \mathbf{X}_2 \mathbf{X}_1^{-\frac{1}{2}}$.

On the tangent space, the Euclidean distance

$$d^2_{\mathcal{E}}(\mathbf{X}_1, \mathbf{X}_2) = \mathrm{tr}[(\mathbf{X}_1 - \mathbf{X}_2)^2], \tag{6.21}$$

with $\mathbf{X}_1 = \log_{\mathbf{Y}} \mathbf{X}_1$ and $\mathbf{X}_2 = \log_{\mathbf{Y}} \mathbf{X}_2$ for any $\mathbf{Y} \in Sym_d^+$, is the first-order approximation of Eq. (6.20).

In [TPM08], a boosting framework on Sym_d^+ for detection (*i.e.*, binary classification) is presented. The idea is to build weak learners by regression over the mappings of the training points on a suitable tangent plane. This tangent plane is defined over the weighted Karcher mean [Kar77] of the positive training data points, such that they preserve their local layout on Sym_d^+. The negative points (i.e. all but pedestrians) instead are assumed to be spread on the manifold, thus including them in the mean estimation, which would bias the result.

Once moving from binary to multi-class classification the above considerations do not hold any longer, because one could have many "positive" classes, each of them localized in a different part of the manifold. Therefore, 1) choosing the Karcher mean of one class would privilege that class with respect to the others, 2) the Karcher mean of all classes is inadequate.

A thorough analysis of Sym_d^+ opens a new perspective. First, its *sectional curvature*, the natural generalization of the classical Gaussian curvature for surfaces, is non-positive. Since Sym_d^+ is actually a symmetric space, the following formula holds for computing the sectional curvature $\kappa_{\mathbf{I}_d}$ at $\mathbf{I}_d$ – due to the homogeneity of Sym_d^+ [Cha06], there is no loss of generality – with $\mathbf{x}$,$\mathbf{y} \in Sym_d$ linearly independent:

$$\begin{aligned} \kappa_{\mathbf{I}_d}(\mathbf{x}, \mathbf{y}) &= \frac{\langle R(\mathbf{x}, \mathbf{y})\mathbf{x}, \mathbf{y} \rangle}{\| \mathbf{x} \|^2 \| \mathbf{y} \|^2 - \langle \mathbf{x}, \mathbf{y} \rangle^2} = \frac{\mathrm{tr}[[\mathbf{x}, \mathbf{y}], \mathbf{x}]\mathbf{y})}{\mathrm{tr}(\mathbf{x}^2)\,\mathrm{tr}(\mathbf{y}^2) - (\mathrm{tr}(\mathbf{x}\mathbf{y}))^2} = \\ &= 2\,\frac{\mathrm{tr}((\mathbf{x}\mathbf{y})^2 - \mathbf{x}^2\mathbf{y}^2)}{\mathrm{tr}(\mathbf{x}^2)\,\mathrm{tr}(\mathbf{y}^2) - (\mathrm{tr}(\mathbf{x}\mathbf{y}))^2}, \end{aligned} \tag{6.22}$$

by the cyclical property of the trace. Here, $[\mathbf{x}, \mathbf{y}] = \mathbf{x}\mathbf{y} - \mathbf{y}\mathbf{x}$ is the matrix commutator, and $R(\mathbf{x}, \mathbf{y}) : \mathbf{z} \mapsto [[\mathbf{x}, \mathbf{y}], \mathbf{z}]$ is the *Riemann curvature operator* (in the symmetric space framework). It can be shown (for the actual proof, see Appendix in the additional material), that $\kappa_{\mathbf{I}_d}(\mathbf{x}, \mathbf{y}) \leq 0$.

Now, an application of Preissmann's theorem [Cha06] shows that, taking the geodesic triangle with vertices $\mathbf{I}_d$, $\mathbf{X}_1$, $\mathbf{X}_2$, one gets

$$d_{\mathcal{E}}(\log_{\mathbf{I}_d} \mathbf{X}_1, \log_{\mathbf{I}_d} \mathbf{X}_2) \leq d(\mathbf{X}_1, \mathbf{X}_2) \tag{6.23}$$

More precisely,

$$d(\mathbf{X}_1, \mathbf{X}_2) = d_{\mathcal{E}}(\log_{\mathbf{I}_d} \mathbf{X}_1, \log_{\mathbf{I}_d} \mathbf{X}_2) + \Xi(\kappa_{\mathbf{I}_d}) \tag{6.24}$$

where $\Xi(\kappa_{\mathbf{I}_d}) \geq 0$ is a function that depends on the sectional curvature. An explicit form for Ξ cannot be easily derived, but it is evident that if the sectional curvature is "small", one can replace the "true" distance with the Euclidean one.

Notice that the remark above reconciles the present "classical" approach with the one in [Pen04, AFPA05], where the Log-euclidean metric is employed throughout, upon endowing Sym^+ with a Lie group structure.

The reasoning above suggests a practical manoeuvre to check this condition. A representative set of covariance matrices is randomly picked from the datasets under observation and the sectional curvature (Eq. 6.22) is estimated for each pair, calculating the mean at the end. Experimentally, this mean value results -10^{-3}, that is far from the standard negative curvature of -1.

In these conditions, one can choose any point on Sym_d^+ on which to map the dataset, and execute the learning on that (Euclidean) space. In practice, the identity matrix $\mathbf{I}_d$ is chosen, as this simplifies the computation. Indeed, Eq. (6.19) becomes

$$\log_{\mathbf{I}_d}(\mathbf{X}) = \log(\mathbf{X}) = \mathbf{U}\log(\mathbf{D})\mathbf{U}^T, \tag{6.25}$$

where $\mathbf{U}\log(\mathbf{D})\mathbf{U}^T$ is the eigenvalue decomposition of $\mathbf{X}$, with $\mathbf{X}$ a generic point in Sym_d^+, $\mathbf{U}$ an orthogonal matrix, and $\log(\mathbf{D})$ the diagonal matrix composed by the eigenvalues' logarithms.

Moreover, the tangent space is the space of symmetric matrices, but there are only $d(d+1)/2$ independent coefficients, which are the upper triangular or lower triangular part of the matrix. Therefore, by applying the vector operator, an orthonormal coordinate system for the tangent space is defined as in Sec. 2.2.6.

6.3.2.2 Algorithm description

Following the considerations above, one can map the dataset $\mathcal{S}$ to the tangent Euclidean space $T_{\mathbf{I}_d}Sym_d^+$, performing the classification directly on this space. In this way, $\mathcal{S}_T = \{\mathbf{x}_i, y_i\}_{i=1,\dots,N}$ is the mapped dataset, with $\mathbf{x}_i = \mathrm{vec}(\log_{\mathbf{I}_d}(\mathbf{X}_i))$.

The essence of a boosting algorithm (see Sec. 3.2 for details) is an iterative reweighting system that tends to focus on the most difficult examples in the training set. In the multi-class classification there are J different sets of weights built from the posterior distribution. Let $\mathrm{P}_j(\mathbf{x}_i)$ be the posterior probability for a training example $\mathbf{x}_i$ to belong to the j-th class. It is computed as:

$$\underset{j}{\mathrm{P}}(\mathbf{x}_i) = \frac{e^{F_j(\mathbf{x}_i)}}{\sum_{k=1}^{J} e^{F_k(\mathbf{x}_i)}}, \quad F_j(\mathbf{x}_i) = \sum_{l=1}^{L} f_{lj}(\mathbf{x}_i), \tag{6.26}$$

where $\{f_{lj}\}_{l=1,\dots,L}$ is a class-specific set of weak learners. Each example in the training set $\mathcal{S}_T$ is associated with a weight that depends on the class considered:

$$w_{ij} = \underset{j}{\mathrm{P}}(\mathbf{x}_i)(1 - \underset{j}{\mathrm{P}}(\mathbf{x}_i)). \tag{6.27}$$

The core of the learning process is the definition of the inter-class decision boundaries, which are carried out by weak learners. Weak classifiers $g_{lj} : T_{\mathbf{I}_d}Sym_d^+ \to \{-1, 1\}$ are built solving a binary classification problem, one class against the others, then the multi-class classifiers $f_{lj} : T_{\mathbf{I}_d} \mapsto \{1, \dots, L\}$ derive from their combination.

The binary weak learners g_{lj} solve a weighted regression problem, whose goodness of fit is measured by the response values $\mathbf{z}_{ij}$, defined as:

$$\mathbf{z}_{ij} = \frac{y^*_{ij} - \mathrm{P}_j(\mathbf{x}_i)}{\mathrm{P}_j(\mathbf{x}_i)(1 - \mathrm{P}_j(\mathbf{x}_i))}, \tag{6.28}$$

where $y^*_{ij} = 1\{j == y_i\}$ is an indicator function. The combination of a set of J binary weak learners g_{lj} is provided by the following equation [FHT00]:

$$f_{lj}(\mathbf{x}_i) = \frac{J-1}{J}\left(g_{lj}(\mathbf{x}_i) - \frac{1}{J}\sum_{k=1}^{J} g_{lk}(\mathbf{x}_i)\right). \tag{6.29}$$

It should be noted note that this operation is possible because the $g_{lk}(\cdot)$s live in the same domain $T_{\mathbf{I}_d}$. If the binary classification had been carried out mapping each class in a different space, similarly to [TPM08], the combination of the results would have been much more complicated and unclear. Working on $T_{\mathbf{Id}}$ represents an elegant and reasonable solution to the problem.

In the following some details of the algorithm are explained, summed up in pseudo-code 12.

Algorithm 12: Multi-class LogitBoost on Sym_d

Data: $(\mathbf{X}_1, y_1), \ldots, (X_N, y_N)$ with $\mathbf{X}_i \in \mathcal{M}$ e $y_i \in \{1, \ldots, J\}$
Result: the ensemble of classifiers $\{F_1, \ldots, F_J\}$.
begin
  Map the data points to the tangent space $T_{\mathbf{I}_d}$, by $\mathbf{x}_i = \mathrm{vec}(\log_{\mathbf{I}_d}(\mathbf{X}_i))$;
  Start with weights $w_{ij} = 1/N$ and $i = 1, \ldots, N$, $F_j(\mathbf{x}_i) = 0$ e $\mathrm{P}_j(\mathbf{x}_i) = 1/J\ \forall j$;
  for $l = 1, 2, \ldots, L$ **do**
    for $j = 1, 2, \ldots, J$ **do**
      Compute the response values (Eq. 6.27) and weights (Eq. 6.28);
      Fit the function $g_{lj}(\mathbf{x}_i) : \mathbb{R}^m \mapsto \mathbb{R}$ by weighted least-square regression of $\mathbf{z}_{ij}$ to $\mathbf{x}_i$ using weights w_{ij};
      Set $F_j(\mathbf{x}_i) \leftarrow F_j(\mathbf{x}_i) + f_{lj}(\mathbf{x}_i)$ where $f_{lj}(\mathbf{x}_i)$ is defined in Eq. (6.29);
      Update $\mathrm{P}_j(\mathbf{x}_i)$ as in Eq. (6.26);

6.3.2.3 Algorithm details

Binary weak classification strategy. In boosting, it is possible to use very different types of weak learners. The most common are the decision stumps (or regression stumps), which are piecewise constant regression functions or linear regression functions. The original LogitBoost algorithm adopts linear regression functions as proposed in [FHT00]. In a binary classification task a linear regression can be sufficient to solve the problem, as shown in [TPM08] for pedestrian detection. However, a more powerful weak classification strategy is mandatory for the multi-class classification problem, as evidenced in [ZZMC07], where piecewise constant functions are used.

After investigating different solutions, the weighted *regression trees* [Bre84] have been selected. They are more powerful than global models, like linear or

polynomial regressors, where a single predictive formula is supposed to hold over the entire data space. Moreover they have lower computational costs, in both the learning and the testing phases. In order to avoid the risk of overtraining of the regression tree, a stopping rule is established. It consists in a minimal number τ of observations per tree leaf, experimentally estimated (see Sec. 5.3.2).

Stop condition. It is important to specify an automatic stop criterion for the learning phase. The proposed rule is a composition of two terms. The first term takes into account the accuracy with which the classes are correctly classified: the maximum accuracy τ_{acc} for all the classes is fixed. The second term concerns the *learning rate*, which is the difference in accuracy between two consecutive iterations of LogitBoost. If the learning rate is less than τ_{lr} for all the classes, one can assume that the boosting process has converged to its optimal solution. More formally, the learning process is stopped at the l-th iteration, when

$$\underset{l}{\text{acc}}(j) \geq \tau_{\text{acc}} \ \vee \ (\underset{l}{\text{acc}}(j) - \underset{l-1}{\text{acc}}(j)) \leq \tau_{\text{lr}}, \ \forall j \in \{1, \ldots, J\}, \tag{6.30}$$

where $\text{acc}_l(j)$ counts the examples of the j-th class correctly classified at the l-th iteration. In all the experiments, τ_{acc} is set to 99% and τ_{lr} to 1%.

Multi-class detection. The proposed multi-class algorithm can be naturally extended to detection purposes simply by adding a class that contains background examples. It is a very large class, because it is potentially composed of all of the possible images that do not contain foreground examples. For this reason, the LogitBoost classifier is combined with a *rejection cascade structure* [VJ01].

Alg. 12 becomes the learning procedure of each cascade level. The stop condition for a cascade level is given by Eq. (6.30), except for the background class that is optimized to classify at least the 35% of the examples in this class correctly, as in [TPM08]. In practice, the examples in the background (BG) class are ordered, according to $P_{\text{BG}}(\mathbf{x})$. Let $\mathbf{x}_{\text{BG}}$ be the element with $(0.35N_{\text{BG}})$-th smallest probability among all the background examples. $th_k = F_{\text{BG}}(\mathbf{x}_{\text{BG}})$ is fixed, where k is the current cascade level.

At the cascade level $(k+1)$, the BG class is first pruned using the cascade of k classifiers, rejecting the samples classified correctly as background. To obtain the desired rejection rate, the classification response for BG is redefined as $F_{\text{BG}}(\mathbf{x}) = (F_{\text{BG}}(\mathbf{x}) - th_k)$.

Computational considerations. The proposed framework inherits some of the computational characteristics of [TPM08], where the main cost is due to SVD factorization needed for the projection of the covariance matrices on the tangent space (see Eq. 6.25). In our case, the presence of a unique projection point decreases the number of required SVD factorizations. This means a dramatic reduction of the computational cost in both the learning and the testing phase.

6.3.3 Experiments

Here different video surveillance applications are shown, where the proposed framework applies: head pose classification, pedestrian detection, and head detection

+ pose classification. In the first two cases, where comparative tests on shared databases are feasible, the relative best performances in the literature is outperformed. In the third case, only qualitative results can be appreciated.

6.3.3.1 Head pose classification

A multi-class classifier for head pose classification is built on the QMUL 4 Pose Head Database [OGX09]. This dataset contains head images of dimension 50×50, obtained from the i-LIDS dataset [Off08]. These images come from a real video surveillance scene, well mirroring typical critical conditions: they are noisy, motion-blurred, and at low resolution. The images are divided into 4 foreground (FG) classes: Back (4200 examples), Front (3555 examples), Left (3042 examples), and Right (4554 examples). Moreover, this dataset contains another set of 2216 background (BG) images. The FG dataset is partitioned into 2 equal parts, using one partition for training and one for testing. From each image I a set Φ of $d = 12$ features is extracted an is composed of:

$$\Phi = \left[X \; Y \; R \; G \; B \; I_x \; I_y \; O \; \mathrm{Gab}_{\{0,\pi/3,\pi/6,4\pi/3\}} \right]. \tag{6.31}$$

X, Y represent the spatial layout in I, and R, G, B are the color channels. I_x and I_y are the directional derivatives of I, and O is the gradient orientation. Finally, Gab is a set of 4 maps containing the results of Gabor filtering. I would like to stress that these features are particularly suited for head orientation classification. Apart from the general position (X, Y) and shape information (I_x, I_y), the covariance of the color channels allows to detect hair and skin textural properties. This particularly helps in distinguishing frontal from back views. Moreover, Gabor filters emphasize facial details, such as the vertical orientation for the nose, or the horizontal orientation of the mouth, if visible. Different combinations of these filters have been tried, and the best results are obtained with dimension 2×4, sinusoidal frequency 16, and directions $\mathcal{D} = \{0, \pi/3, \pi/6, 4\pi/3\}$. In order to give an idea of how the choice of the features affects the system's performances, Fig. 6.7 depicts the behaviour of the system in terms of mean classification accuracy by considering different subsets of Φ.

Once the features are extracted, the covariance matrices from all the patches of $p \times p$ pixels are calculated, on a fixed grid of $p/2$ pixels steps. This means that the patches remain overlapped by half of their size. p is varied to investigate how the dimension (and, thus, the number) of the patches modifies the classification performances. The best performance is obtained with $p = 0.32s$, where s is the image dimension. As visible in Fig. 6.8, enlarging the patch dimension to more than this value diminishes the accuracy. This highlights that having a high number of small patches is better than having few large ones. This because with less, large-sized covariance matrices, all of the image details are mixed together, losing the spatial information.

For each patch, a 4-class classifier is built, as described in Sec. 6.3.2.2. The τ parameter, that rules the complexity of the regression trees, has been fixed to the optimal value 150, according to the accuracy test in Fig. 6.9. It is interesting to note that exceeding this value, the performance drops, which is a sign of overtraining of the system.

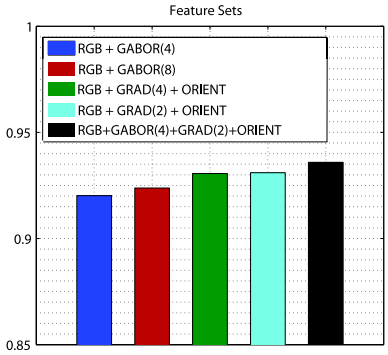


Fig. 6.7. Statistics on the feature vector Φ for the ARCO feature.

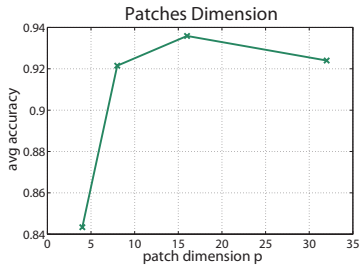


Fig. 6.8. Statistics of the patch dimensions p for the ARCO feature.

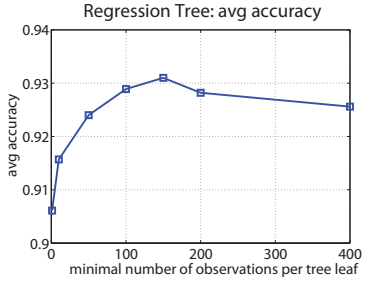


Fig. 6.9. The regression tree stop criterion (the number τ of elements per leaf) for the ARCO feature.

A very important result is the ability to maintain a high classification accuracy on extremely low resolution images. Fig. 6.10 shows the performance of the proposed classifier varying the image dimension s (and changing proportionally the patch parameters, with $p = \lceil 0.32s \rceil$). On a 5×5, image the proposed framework reaches an average accuracy above 82%.

Moreover, the ability of the proposed classifier to deal with occlusions is tested. Indeed, patch-based classifiers, as part-based classifiers, are naturally able to manage the presence of occlusions. In Fig. 6.11 the robustness to four types of occlu-

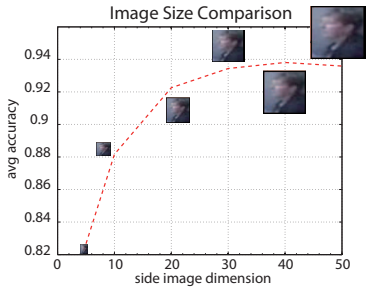


Fig. 6.10. The test image dimensions used for the ARCO feature.

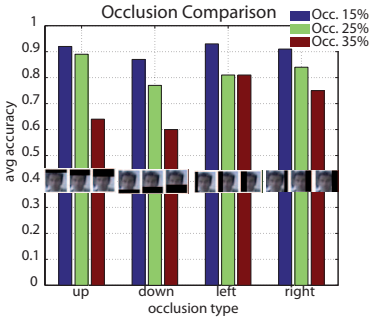


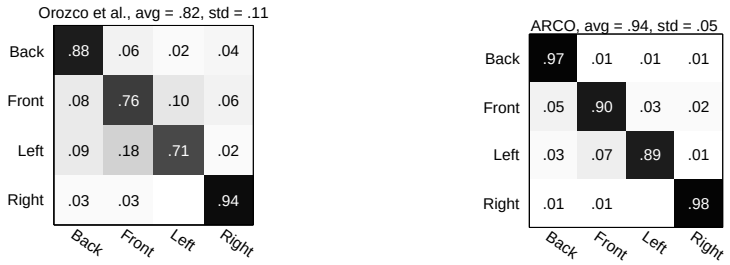
Fig. 6.11. Occlusions of different strength for the ARCO feature.

sions (left-, right-, top- and bottom-side) is depicted, in different sizes. As visible, top and bottom occlusions reduce the performances more, because they completely hide meaningful parts of the face.

Last, the proposed framework is compared with Orozco et al. [OGX09], the state-of-the-art method for head pose classification for low resolution data. It is a head pose descriptor based on similarity distance maps to mean appearance templates of head images at different poses. All images in this dataset have their related pose descriptors, provided by the authors themselves [OGX09]. The classifier is trained by Support Vector Machines (SVMs) using a polynomial kernel, as done in [OGX09]. The result of the comparison, in terms of confusion matrix, is reported in Fig. 6.12. The average rate is 93.5% for the proposed model, against 82.3% for Orozco's model.

6.3.3.2 Pedestrian detection

The proposed framework is instantiated on the binary problem of pedestrian detection to verify the performance of the proposed approach on a pure detection task. The INRIA Person dataset is adapted [DT05] for testing. It contains 1212 human images for the training part of dimension 128×64 and 1133 images for the testing part. A region of interest of 50×50 at the center of the pedestrian images, that corresponds to the actual region where the pedestrian is enclosed (all positive



(a) (b)

Fig. 6.12. In (a) the confusion matrix for the method proposed in [OGX09] and in (b) the confusion matrix associated with ARCO for head orientation classification task.

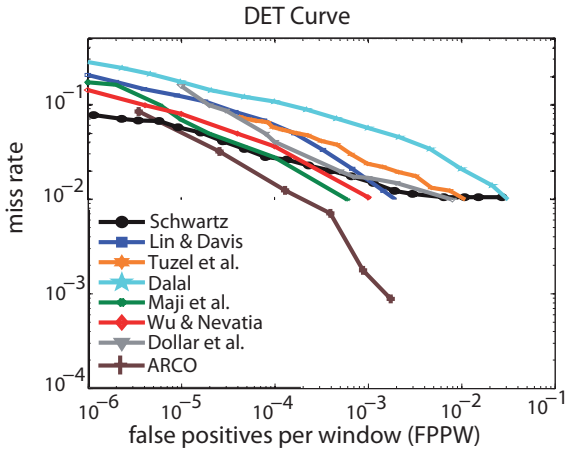


Fig. 6.13. DET curve for pedestrian detection, compared with the state-of-the-art methods [TPM08, DT05, SKHD09, MBM08, LD08b, WN05, DBB$^+$08].

examples come with a quite large border) is picked. Then, the same patch configuration described above (Sec. 6.3.3.1) is used, but with a set of features Φ more suitable for the detection task, i.e. the same proposed in [TPM08]. In Fig. 6.13, the proposed framework is compared with [TPM08] and with the methods in [DT05, SKHD09, MBM08, LD08b, WN05, DBB$^+$08]. The performances are evaluated by the Detection Error Tradeoff (DET) curve, that expresses the proportion of true detections against the proportion of false positives, on a log-log scale. The curve is estimated by varying the threshold th_k in the range $[-1, 1]$. A rejection cascade of 5 levels in which each level is populated by 10000 background examples has been applied. Augmenting the number of cascade levels to more than 5 does not appreciably increase the accuracy, since the number of covariance features remains fixed (in [TPM08], instead, at each step a new feature is selected). This detector clearly outperforms the other methods at the state-of-the-art, especially in terms of miss-rate.

In Fig. 6.14 a comparison on the INRIA Person dataset between the approach described in Sec. 5.2, the approach presented in Sec. 5.3, and the one presented in this Section is made.

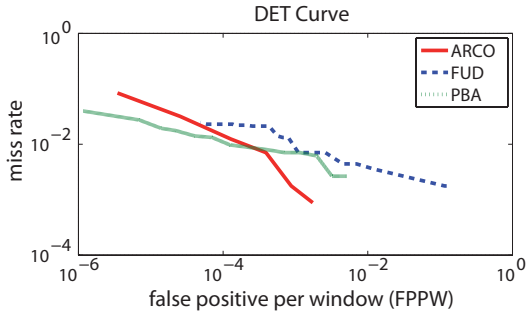


Fig. 6.14. Comparison between the FUD approach (described in Sec. 5.2), the PBA approach (presented in Sec. 5.3), and the one presented in this Section (Sec. 6.3).

6.3.3.3 Head pose detection and classification

One can simply add a background class to the classification problem at hand to perform detection along with classification. Here, I show how the system works for the problem of head pose detection and classification.

The first experiment the 4 head pose classes of the QMUL head pose dataset is considered. Despite to Sec. 6.3.3.1, 2215 background examples are now added to the classification problem. The same optimal settings estimated above is used, and the performance of the proposed approach with [OGX09] are compared. Even though the original paper performs classification only, so the comparison is a bit unfair, its template descriptor is provided for background images also. The background class is added to the other positive classes, and the classification is computed by using SVMs, as described in the paper. The comparison, shown in Fig. 6.15, shows the ability of our system to deal naturally with this task also.

On the other hand, the images of this dataset, though challenging for location and scale variations, are all taken from the same scene, with scarce lighting variations. So, the trained model is not general enough to work with different scenarios. For this reason, a second experiment is performed, building another model, and enriching the training set with new data coming from a different, more general, dataset. The head dataset employed in [LZHT08] is used, composed of 2736 20×20 head images, contained in a ROI of 32×32 pixels. This dataset is mostly obtained from the INRIA person dataset, thus the images are taken from many different scenes and with a large variation of illumination conditions. The set of negative examples is composed by different real scenarios and other images containing body parts. The data is organized in 4 classes (plus background) according to head orientation, since the original dataset does not contain such information.

The positive examples from the 4 Pose Head dataset are resized to 20×20 pixels, whereas for the other dataset the examples are cropped from the center of the ROI.

Half of data are used for training, and the testing set is just composed of the testing set of [LZHT08]. Fig. 6.15(c) summarizes the detection and classification results. Due to the variations in scale and position of the head, the cropped images can contain the head only partially. This is not a problem, though, since the proposed model is robust to partial occlusions, as shown before.

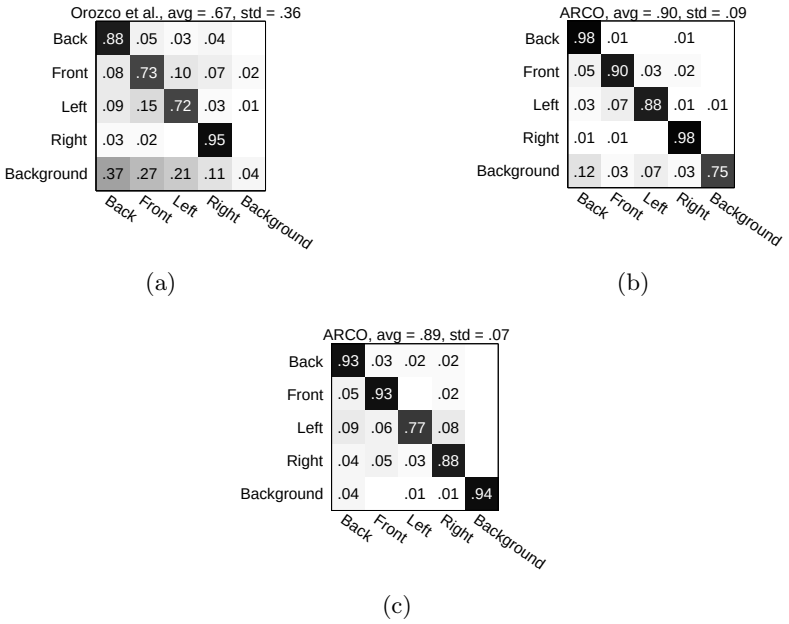


Fig. 6.15. Confusion matrices for the experiments on head pose detection and classification. In (a) e (b), results for the first experiment on 5 Classes QMUL dataset (Orozco's method in (a) [OGX09], ARCO in (b)). (c) is the result of the second experiment with the more general dataset (see text for details).

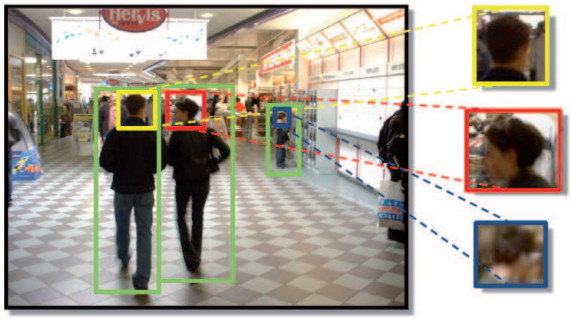
6.4 WARCO (Weighted ARray of COvariance) Matrices

In this Section, a significant revision and extension of ARCO, called WARCO, is presented. It revisits this feature reporting a comprehensive theoretical analysis that motivates some fundamental choices with regard how it is possible to compute the distance among covariance matrices. Moreover, the study goes a step further proposing different approximations of that distance and showing the goodness of this framework in both theoretical and empirical ways. Moreover, with WARCO, a more effective and efficient statistical framework is introduced, if compared to the one proposed in Sec. 6.3. A thorough evaluation is finally provided, on several public datasets, specifically devoted to head orientation classification, human body pose classification, and head orientation estimation in real surveillance scenarios,

showing that the proposed method outperforms in most of the cases the state-of-the-art results.

In computer vision, especially in video surveillance, the capability of characterizing humans is surely of primary importance. With regard to this, social signal processing studies [VPB09] support the hypothesis that the body appearance is critical to infer many behavioural traits, yielding to fine activity profiles. For example, head direction is fundamental to discover the attention focus of individuals [SBOGP08, RR11] and to detect interacting people [CBP+11], while body posture and gestures during an interaction are typically indicators of the speaking activity [CPV+11].

Characterizing humans is particularly troublesome when small and noisy images are handled. In such cases, tasks as body or head orientation estimation (see Fig. 6.16(a)) become serious challenges. This fact induced researchers to design novel features, such as robust classifiers or regressors, to exploit the available small bunch of pixels at best.



(a)

Fig. 6.16. Example of an image from a video surveillance sequence, containing pedestrians and close-up of their heads.

Recently, the use of covariance descriptors as composite features emerged as a powerful means for pedestrian detection [TPM08]. In general, covariances showed to be naturally suited for encoding classes of objects with high intra-class variation, exploiting it to encode systematically mutual relations among basic cues (as gradient, pixel intensity, etc.) [TPM06, DB08, YO08, WN08]. For the pedestrian case, Tuzel et al. [TPM08] employed a boosting framework on Sym_d^+, namely the set of positive definite $d \times d$ symmetric matrices (covariance matrices). The idea was to build weak learners by regression over the mappings of the training points on a suitable tangent space, which was defined over the weighted Karcher mean [Kar77] of the positive training data points, so that they preserve their local layout on Sym_d^+. The negative points (i.e. all but pedestrians), instead, were assumed to be spread on the manifold, without including them in the estimation of the mean.

My aim is to move to a multi-class classification scenario, considering head and body orientations as object classes. In such a scenario, the above-mentioned considerations do not hold any more, because many "positive" classes are given,

each of them localized in a different part of the manifold. As a consequence, 1) choosing the Karcher mean of one class would privilege that class with respect to the others, and 2) the Karcher mean of all classes is inadequate. Therefore, my first contribution consists in a theoretical analysis of this space, to derive a point individuating a common *suitable* projection point that do not penalize any class. Such a point is chosen by analysing the local geometry of the manifold of the considered samples, realizing that, whenever the (sectional) curvature of the manifold is in general weak, a good candidate is the identity. This allows to consider covariance matrices as vectors in a Euclidean space where state-of-the-art classifiers can be utilized.

The second contribution consists in providing a novel measure to calculate the distances between the projected points, preserving the original geodesic distance robustly and in a finer way, if compared to adoption of the Euclidean distance. This comes by considering the sectional curvature of the manifold and adopting the general Campbell-Baker-Hausdorff (CBH) expansion [DK00].

In order to give a rough idea of it, and working with (square) matrices, CBH stems from the elementary fact that, since $\mathbf{X}$ and $\mathbf{Y}$ do not commute in general, one also has $\exp\mathbf{X} \cdot \exp\mathbf{Y} \neq \exp\mathbf{Y} \cdot \exp\mathbf{X} \neq \exp(\mathbf{X}+\mathbf{Y})$. Hence, the CBH-formula, valid in any Lie algebra, is given as a series expansion in terms of nested commutators, of the following form:

$$\exp\mathbf{X}\cdot\exp\mathbf{Y} = \exp(\mathbf{X}+\mathbf{Y}+\frac{1}{2}[\mathbf{X},\mathbf{Y}]+\frac{1}{12}[\mathbf{X},[\mathbf{X},\mathbf{Y}]]+\frac{1}{12}[\mathbf{Y},[\mathbf{Y},\mathbf{X}]]+\cdots). \quad (6.32)$$

The CBH expansion allows to detect the role of the curvature of the manifold, showing that the higher the curvature, the rougher the approximation of the distance. At the same time, our formulation provides a new approximation for the genuine geodesic distance on the manifold, finer than the Euclidean distance previously adopted in Sec. 6.3. It is dubbed such an approximation CBH1, i.e. obtained by exploiting the first term of the CBH expansion.

As third contribution, a novel object descriptor is proposed, expressively designed for encoding complex objects as pedestrians captured by few noisy pixels. The resulting descriptor is dubbed Weighted ARray of COvariances (WARCO), composed of a variable number of overlapped squared patches, each of them described by a covariance matrix of image features. Each covariance is fed into a local weighted classifier (a kernel classifier), where the weight - learned during the training stage - highlights its ability in encoding a defined portion of the object of interest. All the local classifiers are then combined linearly in a strong global classifier.

Adopting WARCO in the proposed theoretical framework allows to build robust kernel classifiers in a very economical way, since the building of the Gram-matrix turns out to be linear in the number of training examples as compared with the quadratic complexity in case of the (exact) geodesic distance.

A thorough experimental section on head orientation classification/regression and body orientation classification promotes the proposed approach as a basic module for advanced surveillance, when fine analyses have to be carried out in difficult scenarios. In particular, it is tested on six different benchmark datasets (including QMUL head dataset, IDIAP head pose dataset, CAVIAR), proposing

three novel sets for head and body orientation estimation. Excellent results are obtained in all the cases.

The rest of the section is organized as follows. In Sec. 6.4.1, the related literature is reported evidencing the novel aspects of my proposal. In Sec. 6.4.2, the mathematical analysis of Sym_d^+ is presented, which has produced interesting theoretical findings exploited to design the statistical method. In Sec. 6.4.3, the kernel-based classification model is describe, which is extensively tested using several public datasets, whose results are illustrated in Sec. 6.4.4.

6.4.1 Related Work

Here the attention is focused on models, object representations and features for robust human body parts description and classification. In this context, the methods can be categorised in general-purpose (e.g. [DT05, VZ10, GL09, TPM06]) and task-specific models (e.g.[FGMR10, WN09, DTPB09, EG09, OB07, ARS09, OGX09]).

As for the task-specific models, two tasks are considered: head and body orientation classification. Several successful human descriptors have been derived in the context of the pedestrian detection problem. Typically, they represent a human as a set of unsupervised selected parts [FGMR10, WN09, DTPB09, SM07, TPM08, MYL$^+$08, WS08, MCT09, WHY10, WMSS10, BHLKG10], where such parts are represented by dense features such as Haar-wavelet-based descriptors, Shapelet [SM07], covariance matrices [TPM08], part-templates [LD08b], Joint Ranking of Granules (JRoG) [WHY10], Local Binary Patterns (LBP) [MYL$^+$08, WHY10], combination of HOG [FGMR10], Integral Channel Features [DTPB09], self-similarity on color channels [WMSS10], and synthesized features [BHLKG10]. Other works combine some of the above-mentioned features as [WHY10], where HOG and LBP are concatenated, and [WS08], where HOG, Haar-like, and Shapelet features are used. Most of these approaches use boosting both for a greedy estimation of the most discriminative patches and classifying them at the same time. A relevant exception is [FGMR10], which presents a part-based deformable model for object detection. Considering HOG features [DT05], the object model is defined by a constellation of discriminative learned parts that score subwindows of a ROI (Region Of Interest) containing the OI, and the classification framework is represented by latent Support Vector Machines (SVMs).

There is also a large literature concerning the head orientation estimation task, [MCT09, SBB$^+$09, BO05, FGVG11, HSDlTB11]. For high resolution images, important methods are proposed in the context of the CLEAR07 challenge [BO05]. Instead, for low resolution images, the head orientation estimation task often translates into the head orientation classification task, in which there are few works in the state of the art. Two recent approaches [RR06, OGX09] provide valid solutions to these problems. Both works organize the overall processing scheme into two phases: detection and categorization.

Similarly to the head orientation estimation, for the human pose estimation task there are many methods considering high-resolution images [AT06, ARS09, BM09, TF10]. Few methods can deal with small pedestrians, classifying their body orientation. An interesting example is [EG09], where a coarse-to-fine matching of an exemplar-based shape hierarchy and Chamfer distance are used to find the best template describing a candidate human orientation.

Considering general purpose models, probably the most important example is the detector proposed by Dalal & Triggs [DT05]. This detector, which uses the HOG as feature, still represents an effective solution to the object detection and classification tasks. HOG describes an object as a fine set of overlapped blocks and the algorithms utilize a sliding window procedure, where a discriminative SVM model is applied to all positions and scales of an image. This approach has been used also in [AT06] for human pose estimation, which is recovered by direct regression of the HOG descriptors. Moreover, Agarwal & Triggs have demonstrated an application of non-negative matrix factorization that allows to discriminate features of interest from background. Another interesting approach based on HOG features is proposed by Lin & Davis [LD08b]. It adopts an OI model similar to the one proposed in this Section. In fact, instead of standard concatenation-style image location-based feature encoding, patches are evaluated independently and then a probabilistic framework is used to link the evaluation results. Some years later another successful work has been proposed by Schwartz er al. [SKHD09]. It uses HOG features again, on both colour and gray scale images, and the pre-process the feature space using partial least squares to reduce its dimensionality.

Recently, in [EG10], the HOG representation has been employed to categorize the pedestrian orientations into few classes, considering pedestrians at low resolution. Moreover, in this case HOG is combined to adaptive local receptive field features in a multi-layer neural network architecture.

In [VZ10], a different kind of histogram-based representation is used, based on the spatial pyramid concept [LSP06]. These two models generalize the previous ones because a multi-layer analysis is performed, but a regular grid structure is still used to represent the object.

A different approach is used in [GL09], where patches are sampled randomly from images to build the object class model using Hough Forest, which is a Random Forest that maps the image patch appearance directly to the probabilistic vote about the possible location of the object centroid, similarly to the implicit shape model. Since fixed size patches are used, the method is adaptable to a wide range of tasks.

The type of OI descriptors, presented in the current section, has been already exploited in the case of pedestrian detection [TPM08],[YO08], and, previously, also in the biomedical research domain [FLPJ04],[FPAA07]. A mathematical derivation is reported in [AFPA08], but the investigation of the properties of covariance matrices as objects living in a non Euclidean space is still an active research topic, due to their versatility and effectiveness when used as descriptors for classification tasks [FVJ08],[SLHN10].

The proposed approach can be categorized as general purpose, and a former version is presented in Sec. 6.3. It differs in several ways as a new weighted covariance descriptor is introduced, which is then exploited adopting a kernel machine architecture suitable both for classification and regression tasks. Moreover, the theoretical part is consistently new. In fact, a rigorous and comprehensive mathematical analysis of the covariance matrices living in a Riemannian manifold, whose findings are utilized to justify and lay down the ground of the proposed statistical classification method.

6.4.2 Theoretical analysis of Sym_d^+

In this Section, basic differential geometry notions about Sym_d^+ are gathered, namely the set of positive definite $d \times d$ symmetric matrices (covariance matrices), adopting the formalism of [Cha06, DK00]; this coverage will allow to introduce my main theoretical contribution, i.e. the application of the Campbell-Baker-Hausdorff expansion as a fast way to approximate distances in Sym_d^+. In particular, after recalling some preliminaries notions in Sec. 6.4.2.1, the fact that Sym_d^+ is a homogeneous space is shown (Sec. 6.4.2.2): this means that one is entitled to select any point on Sym_d^+ to define a tangent space over which projecting points and calculating distances.

In Sec. 6.4.2.3, the fact that the identity $\mathbf{I}_d$ on Sym_d^+ is a particularly convenient choice (under a pure computational complexity aspect) as a projection point. In Sec. 6.4.2.4, the (sectional) curvature of Sym_d^+ is introduced which allows to measure how much Sym_d^+ differs from a Euclidean space, which is flat. In particular, it turns out that Sym_d^+ has a negative curvature, and this will for instance ensure that there is only one geodesic connecting any two points; moreover, this will show that the first correction to the Euclidean distance provided by the CBH-expansion, i.e. our distance approximation, is non negative. This is finally discussed in Sec. 6.4.2.5.

6.4.2.1 Preliminaries

In general, given a Lie group G and a closed Lie subgroup H thereof, the quotient set G/H consisting of all left cosets $[g] := gH = \{g\,h \mid h \in H\}$ becomes in a unique way a smooth manifold (this is the prototype of a G-homogeneous space). The study of the geometrical properties of homogeneous spaces is greatly eased by the fact that all points can be treated on the same footing (colloquially, the manifold appears to be the same when looked upon from whatever point therein). This is quite important, from a machine learning point of view. Therefore, both for theoretical and practical reasons, focusing the attention on the class $[e] = H$ of the neutral element $e \in G$ is natural. A graphical example of homogeneous space is shown in Fig. 6.17.

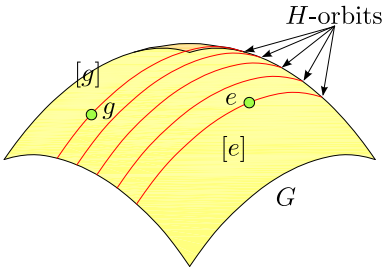


Fig. 6.17. Homogeneous spaces.

6.4.2.2 $\mathbf{Sym_d^+}$ is an homogeneous space.

The general linear group $Gl(d,\mathbb{R})$, consisting of all non-singular real $d\times d$ matrices, naturally acts on Sym_d^+ via *congruence*:

$$Sym_d^+ \ni \mathbf{X} \mapsto \mathbf{M}^T\mathbf{X}\mathbf{M} \in Sym_d^+, \mathbf{M} \in Gl(d,\mathbb{R}). \tag{6.33}$$

By virtue of (a corollary of) Sylvester's theorem [Ser93], the latter action is transitive: in other words, any two positive definite symmetric matrices are congruent, i.e., there is always an $\mathbf{M}$ that connects them. In particular, every matrix $\mathbf{X} \in Sym_d^+$ is congruent to $\mathbf{I}_d$ (the $d \times d$ identity matrix):

$$\mathbf{X} = \mathbf{M}^T \cdot \mathbf{I}_d \cdot \mathbf{M} = \mathbf{M}^T\mathbf{M} \tag{6.34}$$

for some $\mathbf{M} \in Gl(d,\mathbb{R})$; in this scenario one shall take, for specific calculations, $\mathbf{M} = \mathbf{X}^{\frac{1}{2}}$. Therefore, Sym_d^+ is the space of all symmetric matrices congruent to $\mathbf{I}_d$. Also, $\mathbf{I}_d$ is invariant under congruence, namely $\mathbf{M}^T\mathbf{M} = \mathbf{I}_d$, if and only if $\mathbf{M} \in O(d,\mathbb{R})$, the group of orthogonal $d \times d$ matrices. In other words, $O(d,\mathbb{R})$ is the *isotropy group* of $\mathbf{I}_d$. From this, one finds that Sym_d^+ is the homogeneous space

$$Sym_d^+ \cong Gl(d,\mathbb{R})/O(d,\mathbb{R}) \cong Gl_+(d,\mathbb{R})/SO(d,\mathbb{R}) \tag{6.35}$$

(one may restrict to matrices with positive determinant to get connected groups). $SO(d,\mathbb{R})$ denotes the special orthogonal group, i.e. the orthogonal matrices having determinant $+1$. In view of the homogeneity, one can choose to work at the identity, since this will ease all subsequent computations.

6.4.2.3 A Riemannian metric on $\mathbf{Sym_d^+}$

Recall that a Riemannian manifold $(\mathcal{M}, \langle\,,\rangle)$ is a smooth manifold equipped with a Riemannian metric $\langle\,,\rangle$, i.e. a smoothly varying inner product $\langle\,,\rangle_{\mathbf{P}}$ on its tangent spaces $T_{\mathbf{P}}\mathcal{M}$, $\mathbf{P} \in \mathcal{M}$. The tangent vectors (the elements of $T_{\mathbf{P}}\mathcal{M}$) are the "velocities" of the curves in $\mathcal{M}$ issuing from $\mathbf{P} \in \mathcal{M}$ or, equivalently, the "directional derivatives" of the smooth functions defined in a neighbourhood of $\mathbf{P}$.

The tangent space of Sym_d^+ at any point $\mathbf{X}$ (notation: $T_{\mathbf{X}}Sym_d^+$), is Sym_d, the space of symmetric matrices. By homogeneity it is enough to check this at the identity $\mathbf{I}_d$. Indeed, let us consider an interval $J \subset R$ containing 0, and let us consider a smooth curve of matrices $J \ni t \mapsto \mathbf{X}(t) \in Sym_d^+$ with $\mathbf{X}(0) = \mathbf{I}_d$. Its "velocity" at $\mathbf{I}_d$, namely $\dot{\mathbf{X}}(0)$, belongs to Sym_d, since the derivative of $\mathbf{X}(t)$ is still a symmetric matrix. Vice versa, given a matrix $\mathbf{W} \in Sym_d$, it is possible to find a curve in Sym_d^+ starting at $\mathbf{I}_d$ with velocity given by $\mathbf{W} = \dot{\mathbf{X}}(0)$. Taking for instance $\mathbf{X}(t) = \exp(t\mathbf{W})$, if the matrix $\mathbf{W}$ is diagonalized and denote its eigenvalues by w_i, $i = 1, 2, \ldots, d$, then the eigenvalues of $\mathbf{X}(t)$ are $\exp(tw_i) > 0$, $i = 1, 2, \ldots, d$. Therefore, the matrix is positive definite. By continuity, any curve with the same velocity at $\mathbf{I}_d$ is locally in Sym_d^+. Given $\varphi \equiv \varphi_M : Sym_d^+ \ni \mathbf{X} \mapsto \mathbf{M}^T\mathbf{X}\mathbf{M} \in Sym_d^+$, its differential φ_* is:

$$\varphi_* : T_{\mathbf{I}_d}Sym_d^+ \to T_{\mathbf{X}}Sym_d^+, \quad \mathbf{W}' \mapsto \mathbf{M}^T\mathbf{W}'\mathbf{M} =: \mathbf{W}, \tag{6.36}$$

since in general $\partial(\mathbf{M}^T\mathbf{X}\mathbf{M}) = \mathbf{M}^T\partial\mathbf{X}\mathbf{M}$ (here ∂ denotes of course differentiation; notice that $\mathbf{W}$ is symmetric as well, as asserted before). On $T_{\mathbf{I}_d}Sym_d^+$ it is possible to define the Frobenius inner product:

$$\langle \mathbf{W}_1', \mathbf{W}_2' \rangle_{\mathbf{I}_d} := \mathrm{tr}(\mathbf{W}_1'\mathbf{W}_2'), \tag{6.37}$$

where $\mathbf{W}_i' \in Sym_d$. It is extended to a *Riemannian metric, invariant under congruence* via the formula:

$$\langle \mathbf{W}_1, \mathbf{W}_2 \rangle_{\varphi(\mathbf{I}_d)=\mathbf{X}} := \langle \varphi_*^{-1}(\mathbf{W}_1), \varphi_*^{-1}(\mathbf{W}_2) \rangle_{\mathbf{I}_d}, \tag{6.38}$$

namely (by a short computation using $\mathbf{X} = \mathbf{M}^T\mathbf{M}$)

$$\langle \mathbf{W}_1, \mathbf{W}_2 \rangle_{\mathbf{X}} = \mathrm{tr}(\mathbf{X}^{-1}\mathbf{W}_1\mathbf{X}^{-1}\mathbf{W}_2), \tag{6.39}$$

where $\mathbf{W}_1, \mathbf{W}_2 \in T_{\mathbf{X}}Sym_d^+ \cong Sym_d$. This turns out to be well-defined since the Frobenius inner product is $O(d,\mathbb{R})$-invariant: indeed, if $O \in O(d,R)$, one has:

$$\mathrm{tr}(\mathbf{O}^T\mathbf{W}_1'\mathbf{O}\cdot\mathbf{O}^T\mathbf{W}_2'\mathbf{O}) = \mathrm{tr}(\mathbf{O}^T\mathbf{W}_1'\,\mathbf{W}_2'\mathbf{O}) = \mathrm{tr}(\mathbf{W}_1'\mathbf{W}_2'\mathbf{O}\mathbf{O}^T) = \mathrm{tr}(\mathbf{W}_1'\mathbf{W}_2') \tag{6.40}$$

This further entails that, given any two points $\mathbf{X}_1$, $\mathbf{X}_2 \in Sym_d^+$, and

$$\varphi \equiv \varphi_M : Sym_d^+ \ni \mathbf{X} \mapsto \mathbf{M}^T\mathbf{X}\mathbf{M} \in Sym_d^+,$$

then

$$d(\varphi(\mathbf{X}_1), \varphi(\mathbf{X}_2)) = d(\mathbf{X}_1, \mathbf{X}_2), \tag{6.41}$$

where d is the distance induced by the above Riemannian metric (and equals the length of a minimal geodesic connecting the two points - in our case the latter exists and it is unique, see also below); in other words, φ is an *isometry*. In particular, one may compute all distances from a fixed point, the natural choice thereof being the identity. Also, any $\mathbf{X} \in Sym_d^+$ is of the form

$$\mathbf{X} = \exp \mathbf{W}_{\mathbf{X}}, \quad \mathbf{W}_{\mathbf{X}} = \log \mathbf{X} \in Sym_d \tag{6.42}$$

(spectral theorem), therefore

$$d^2(\mathbf{I}_d, \mathbf{X}) = \| \log \mathbf{X} \|^2 = \mathrm{tr}(\log \mathbf{X})^2) = \sum_{i=1}^{d} (\log \sigma_i)^2. \tag{6.43}$$

The σ_i's are the (positive) eigenvalues of $\mathbf{X}$ and, in general (setting below $\mathbf{M}_1^T\mathbf{M}_1 = \mathbf{X}_1$, and specifically $\mathbf{M}_1 = \mathbf{X}_1^{\frac{1}{2}}$):

$$\begin{aligned} d^2(\mathbf{X}_1, \mathbf{X}_2) &= d^2(\varphi_{\mathbf{M}_1}(\mathbf{I}_d), \mathbf{X}_2) = d^2(\mathbf{I}_d, \varphi_{\mathbf{M}_1}^{-1}(\mathbf{X}_2)) \\ &= \mathrm{tr}(\log(\mathbf{X}_1^{-\frac{1}{2}}\mathbf{X}_2\mathbf{X}_1^{-\frac{1}{2}}))^2) = \sum_{i=1}^{d} (\log \xi_i)^2, \end{aligned} \tag{6.44}$$

where the ξ_i's are the (positive) eigenvalues of $\mathbf{X}_1^{-\frac{1}{2}}\mathbf{X}_2\mathbf{X}_1^{-\frac{1}{2}}$. In fact Sym_d^+ is actually a *Riemannian symmetric space* $(\mathcal{M}, \langle , \rangle)$, namely, for each point $\mathbf{P} \in \mathcal{M}$,

there exists an isometry $\sigma_{\mathbf{P}}$ fulfilling $\sigma_{\mathbf{P}}^2 = \mathbf{I}_d\mathcal{M}$ (with $\mathbf{I}_d\mathcal{M}$ the trivial isometry on $\mathcal{M}$) and having $\mathbf{P}$ as an isolated fixed point ([Cha06]). One shall not delve any further into the general theory of symmetric spaces, confining ourselves to recalling specific facts when needed. For example, it follows from it that the geodesics starting from $\mathbf{I}_d$ are of the form

$$\mathbb{R} \ni t \mapsto \exp(t\mathbf{W}) \in Sym_d^+, \ \mathbf{W} \in Sym_d, \tag{6.45}$$

with exp the standard matrix exponential (since, for symmetric spaces associated to matrix groups, the Riemannian exponential coincides, at the identity, with the matrix one). An intuitive pictorial idea of the exponential map is illustrated in Fig. 6.18. In our case, the isometry $\sigma_{\mathbf{P}}$ of the general theory is induced at $\mathbf{I}_d$ by

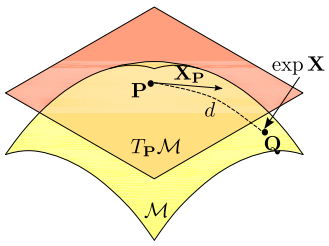


Fig. 6.18. Exponential map.

the map $Sym_d \ni \mathbf{W} \mapsto -\mathbf{W} \in Sym_d$.

6.4.2.4 Non-positivity of the sectional curvature of $\mathbf{Sym_d^+}$

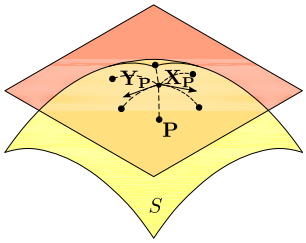
Given a Riemannian manifold $(\mathcal{M}, \langle , \rangle)$ its *sectional curvature* $\kappa_{\mathbf{P}}(\mathbf{X_P}, \mathbf{Y_P})$ at $\mathbf{P} \in \mathcal{M}$, if $\mathbf{X_P}$ and $\mathbf{Y_P}$ are linearly independent tangent vectors at $\mathbf{P}$, is given by

$$\kappa_{\mathbf{P}}(\mathbf{X_P}, \mathbf{Y_P}) := \frac{\langle R(\mathbf{X_P}, \mathbf{Y_P})\mathbf{X_P}, \mathbf{Y_P}\rangle_{\mathbf{P}}}{\langle \mathbf{X_P}, \mathbf{X_P}\rangle_{\mathbf{P}}\langle \mathbf{Y_P}, \mathbf{Y_P}\rangle_{\mathbf{P}} - \langle \mathbf{X_P}, \mathbf{Y_P}\rangle_{\mathbf{P}}^2} \tag{6.46}$$

where R is denoting the Riemann curvature operator (see below). Notice that the denominator represents the area squared of the parallelogram determined by $\mathbf{X_P}$ and $\mathbf{Y_P}$. It is important to pinpoint that the sectional curvature just depends on the plane spanned by $\mathbf{X_P}$ and $\mathbf{Y_P}$, and indeed it turns out to coincide with the Gaussian curvature, at $\mathbf{P}$, of the parametric surface $S : (u, v) \mapsto \exp_{\mathbf{P}}(u\mathbf{X_P} + v\mathbf{Y_P})$ (here $\exp_{\mathbf{P}}$ denotes the Riemannian exponential at $\mathbf{P}$). An example of that is shown in Fig. 6.19.

In geometry, the (sectional) curvature is a measure of non-flatness of the manifold. The local vanishing of the curvature implies that the Riemannian manifold in question is actually a portion of a Euclidean space. One shall exploit this for learning purposes.

a formula for the sectional curvature for Sym_d^+ is worked out, showing that it is non-positive at any point. Since Sym_d^+ is a symmetric space, one can again



(a)

Fig. 6.19. Gaussian curvature ($\kappa_{\mathbf{P}}(\mathbf{X_P}, \mathbf{Y_P})$) of the 2-dimensional surface S at p.

work at the identity, whereat one gets the following expression for the Riemann curvature operator (in the symmetric space framework, see e.g. [Cha06])

$$R(\mathbf{X}, \mathbf{Y}) : Sym_d \ni \mathbf{Z} \mapsto [[\mathbf{X}, \mathbf{Y}], \mathbf{Z}] \in Sym_d. \tag{6.47}$$

Here,

$$[\mathbf{X}, \mathbf{Y}] = \mathbf{XY} - \mathbf{YX}$$

is the matrix commutator. Then the sectional curvature $\kappa_{\mathbf{I}_d}$ at $\mathbf{I}_d$ reads (with $\mathbf{X}$,$Y \in Sym_d$ linearly independent):

$$\kappa_{\mathbf{I}_d}(\mathbf{X}, \mathbf{Y}) = \frac{\langle R(\mathbf{X}, \mathbf{Y})\mathbf{X}, \mathbf{Y}\rangle}{\| \mathbf{X} \|^2 \| \mathbf{Y} \|^2 - \langle X, Y\rangle^2} = 2\frac{\mathrm{tr}((\mathbf{XY})^2 - \mathbf{X}^2\mathbf{Y}^2)}{\mathrm{tr}(\mathbf{X}^2)\,\mathrm{tr}(\mathbf{Y}^2) - (\mathrm{tr}(\mathbf{XY}))^2}, \tag{6.48}$$

by the cyclical property of the trace.

Again, the denominator

$$\| \mathbf{X}_1 \|^2 \| \mathbf{X}_2 \|^2 - \langle \mathbf{X}_1, \mathbf{X}_2 \rangle^2 =: \mathcal{A}(\mathbf{X}_1, \mathbf{X}_2)^2$$

is the area of the parallelogram determined by $\mathbf{X}_1$ and $\mathbf{X}_2$, squared. Therefore, to prove that

$$\kappa_{\mathbf{I}_d}(\mathbf{X}, \mathbf{Y}) \leq 0$$

, it suffices to show that

$$\mathrm{tr}((\mathbf{XY})^2) \leq \mathrm{tr}(\mathbf{X}^2\mathbf{Y}^2), \tag{6.49}$$

and that equality holds if and only if $[\mathbf{X}, \mathbf{Y}] = 0$. This is implied by the following immediate consequence of the Schwarz inequality for (real) inner products

$$\langle \mathbf{x}, \mathbf{y} \rangle \leq \| \mathbf{x} \|^2, \text{ if } \| \mathbf{x} \| = \| \mathbf{y} \| \tag{6.50}$$

(equality holding if and only if $\mathbf{x} = \mathbf{y}$). Indeed, upon setting $\mathbf{x} = \mathbf{XY}$,

$$\mathbf{y} = \mathbf{YX},$$

$$\langle \mathbf{x}, \mathbf{y} \rangle = \mathrm{tr}(\mathbf{x}^T\mathbf{y}) = \mathrm{tr}(\mathbf{y}^T\mathbf{x})$$

, and using $\mathbf{X}^T = \mathbf{X}$, $\mathbf{Y}^T = \mathbf{Y}$, one has:

$$\| \mathbf{y} \|^2 = \mathrm{tr}(\mathbf{YX})^T(\mathbf{YX})) = \mathrm{tr}(\mathbf{X}^T\mathbf{Y}^T\mathbf{YX}) = \mathrm{tr}(\mathbf{XYYX}) = \mathrm{tr}(\mathbf{X}^2\mathbf{Y}^2) = \| \mathbf{x} \|^2 \tag{6.51}$$

As previously anticipated, for learning purposes, $\kappa_{\mathbf{P}}(\mathbf{X_P}, \mathbf{Y_P})$ provides a quantitative measure of how much a Riemannian manifold differs from a flat (i.e. Euclidean) one.

6.4.2.5 An expansion of the distance via the CBH-formula

Recalling that Preismann's theorem (see e.g. [Cha06]) says that any two points of a complete simply connected manifold with non-positive sectional curvature are connected by precisely one geodesic, which is minimizing, given a geodesic triangle with sides of length a, b, c, and angle θ opposite to (the side with length) c, the $a^2 + b^2 - 2ab\cos\theta \leq c^2$ inequality holds. An application of the theorem to Sym_d^+ (which indeed satisfies the above assumptions) shows that, taking the geodesic triangle with vertices $\mathbf{I}_d$, $\mathbf{X}_1$, $\mathbf{X}_2$, one gets $d_{\mathcal{E}}(\log_{\mathbf{I}_d}\mathbf{X}_1, \log_{\mathbf{I}_d}\mathbf{X}_2) \leq d(\mathbf{X}_1, \mathbf{X}_2)$, where $d_{\mathcal{E}}$ denotes the standard Euclidean distance (induced by the Frobenius norm)

$$d^2_{\mathcal{E}}(\mathbf{X}_1, \mathbf{X}_2) = \mathrm{tr}((\mathbf{X}_1 - \mathbf{X}_2)^2) \tag{6.52}$$

with $\mathbf{X}_i = \log_{\mathbf{I}_d}\mathbf{X}_i$. But actually one can easily get approximate formulae for the distance by exploiting the Campbell-Baker-Hausdorff formula (CBH) (see e.g. [DK00], p.30, where the more general Dynkin's formula is given; applying it to the Lie algebra consisting of real $d \times d$ matrices). The crudest approximation beyond the Euclidean distance (computed on the tangent space $T_{\mathbf{I}_d}Sym_d^+$; also set $\mathbf{X}_i := \log \mathbf{X}_i$, $i = 1, 2$) reads:

$$\begin{aligned} d^2(\mathbf{X}_1, \mathbf{X}_2) &= d^2_{\mathcal{E}}(\mathbf{X}_1, \mathbf{X}_2) - \frac{1}{12}\langle R(\mathbf{X}_1, \mathbf{X}_2)\mathbf{X}_1, \mathbf{X}_2\rangle + \cdots \\ &= d^2_{\mathcal{E}}(\mathbf{X}_1, \mathbf{X}_2) - \frac{1}{12}\kappa(\mathbf{X}_1, \mathbf{X}_2) \cdot \mathcal{A}(\mathbf{X}_1, \mathbf{X}_2)^2 + \cdots \end{aligned} \tag{6.53}$$

that it is illustrated in Fig. 6.20.

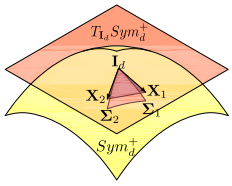


Fig. 6.20. Approximating the true distance.

The calculation employs the CBH-formula (suitably truncated to second order commutators)

$$\log(e^{\mathbf{X}}e^{\mathbf{Y}}) = \mathbf{X} + \mathbf{Y} + \frac{1}{2}[\mathbf{X}, \mathbf{Y}] + \frac{1}{12}[\mathbf{X}, [\mathbf{X}, \mathbf{Y}]] + \frac{1}{12}[\mathbf{Y}, [\mathbf{Y}, \mathbf{X}]] + \cdots \quad (6.54)$$

and it subsequently entails

$$\log(e^{\mathbf{X}}e^{\mathbf{Y}}e^{\mathbf{X}}) = 2\mathbf{X} + \mathbf{Y} - \frac{1}{6}[\mathbf{X}, [\mathbf{X}, \mathbf{Y}]] - \frac{1}{6}[\mathbf{Y}, [\mathbf{X}, \mathbf{Y}]] + \cdots \quad (6.55)$$

The above series are indeed convergent. Upon setting $\mathbf{X} = -\frac{1}{2}\mathbf{X}_1$, $\mathbf{Y} = \mathbf{X}_2$, the r.h.s. of the above formula becomes

$$\mathbf{W} = \mathbf{X}_2 - \mathbf{X}_1 - \frac{1}{24}[\mathbf{X}_1, [\mathbf{X}_1, \mathbf{X}_2]] + \frac{1}{12}[\mathbf{X}_2, [\mathbf{X}_1, \mathbf{X}_2]] + \cdots . \quad (6.56)$$

Now, substituting the above expression in the formula for the distance (Eq. (6.44)), it turns out that, after a short computation exploiting the properties of tr (see Sec. 2.2.3):

$$\begin{aligned} d^2(\mathbf{X}_1, \mathbf{X}_2) = &\operatorname{tr}[(\mathbf{X}_2 - \mathbf{X}_1)^2] - \frac{1}{12}\operatorname{tr}\{[\mathbf{X}_1, [\mathbf{X}_1, \mathbf{X}_2]](\mathbf{X}_2 - \mathbf{X}_1)\} \\ &+ \frac{1}{6}\operatorname{tr}\{[\mathbf{X}_2, [\mathbf{X}_1, \mathbf{X}_2]](\mathbf{X}_2 - \mathbf{X}_1)\} + \cdots \end{aligned} \quad (6.57)$$

The last expression can be eventually transformed into Eq. (6.53) upon recalling the formula for the Riemannian curvature operator (Eq. (6.47)), together with the following general Riemann tensor identities (the third one being the *Bianchi identity*, see e.g. [Cha06]):

$$R(x, y, z, t) = -R(y, x, z, t) = -R(x, y, t, z) = R(z, t, x, y) \quad (6.58)$$

$$R(x, y, z, t) + R(y, z, x, t) + R(z, x, y, t) = 0 \quad (6.59)$$

where $R(x, y, z, t)$ is defined as $\langle R(x, y)z, t\rangle$. In particular, one has

$$R(\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_1, \mathbf{X}_1) = R(\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_2, \mathbf{X}_2) = 0, \quad (6.60)$$

and one can easily get the sought-for approximate formula. In Sec. 6.4.4 the efficacy of the expansion above will be shown [1]. From the classification accuracy point of view, it represent a scalable method to enhance the performance without increasing the computational complexity.

6.4.3 The Statistical Framework

6.4.3.1 The General Architecture

The WARCO classifier has been designed specifically to deal with few visual information, that is, tiny images with noisy pixel values. It consists in a grid of N_P uniformly spaced and overlapped $k \times k$ patches $\Phi = \{\phi_n\}_{n=1,\ldots,N_P}$, where each

[1] The first correction to the Euclidean distance is kept. One could work out more refined expressions upon carefully keeping track of the various summands of CBH expansion. The successive terms, depending on nested commutators, are also related to curvature. Notice that it is not provided precise estimates for the approximation error.

patch is described by a covariance matrix of features. For the sake of generality, neither the degree of overlap nor the nature of the feature considered are not specify here, postponing this aim in the experiments.

In a $L-$class classification scenario, ARCO instantiates an independent classifier on each patch, and provides a joint (log) posterior classification probability which is

$$\mathrm{P}(l|\Phi) = \sum_{n=1}^{N_P} \log(\mathrm{P}(\phi_n)) + \log(\mathrm{P}(\phi_n|l)). \tag{6.61}$$

where $l = 1, ..., L$, $\mathrm{P}(\phi_n|l)$ is a per-patch likelihood probability of the $n-$th classifier and $\mathrm{P}(\phi_n)$ is a normalized weight $\sum_n \mathrm{P}(\phi_n) = 1$ that acts as a prior. This latter has been learned during the training stage, mirroring the reliability of each particular patch in giving the right classification score.

In a regression scenario, WARCO instantiates a regressor for each patch, and the final output is the median of all the outputs of the single regressors.

Standard linear Support Vector Machine (SVM) is the tool employed for performing classification and regression, where the Gram-matrix has been calculated by employing three different distances, i.e.,

$d_{\mathcal{E}}$: The distance between covariance matrices based on the Frobenius norm (see Sec. 6.4.2.3)

$$d^2_{\mathcal{E}}(\mathbf{X}, \mathbf{Y}) = \mathrm{tr}(\log_{\mathbf{I}_d}(\mathbf{X}) - \log_{\mathbf{I}_d}(\mathbf{Y}))^2). \tag{6.62}$$

$d_{\mathcal{CBH}1}$ The distance[2] between covariance matrices exploiting the CBH expansion limited to the first order (see Sec. 6.4.2.5)

$$d^2_{\mathcal{CBH}1}(\mathbf{X}, \mathbf{Y}) = d^2_{\mathcal{E}}(\mathbf{X}, \mathbf{Y}) + \widetilde{\Xi}(\kappa_{\mathbf{I}_d}), \tag{6.63}$$

where $\widetilde{\Xi}(\kappa_{\mathbf{I}_d}) = -\frac{1}{12}\langle R(\log_{\mathbf{I}_d}(\mathbf{X}), \log_{\mathbf{I}_d}(\mathbf{Y}))\log_{\mathbf{I}_d}(\mathbf{X}), \log_{\mathbf{I}_d}(\mathbf{Y})\rangle$.

$d_{\mathcal{G}}$: The actual geodesic distance between covariance matrices (see Sec. 6.4.2.3)

$$d^2_{\mathcal{G}}(\mathbf{X}, \mathbf{Y}) = \mathrm{tr}(\log^2_{\mathbf{I}_d}(\mathbf{X}^{-\frac{1}{2}} Y \mathbf{X}^{-\frac{1}{2}})). \tag{6.64}$$

Given the dissimilarity matrix D built for each of these three distances, one needs to resort to similarity relations, so that a nonlinear transformation of its entries is applied: $\exp(-1/\mu(D)D)$, where $-1/\mu(D)$ is a regularization terms in which $\mu(D)$ is the average value of D. In the computation of D, the logarithmic projection is the most time-consuming operation. Looking at the three distances, one can immediately calculate the number of logarithmic projections needed, which is linear in (6.62) and (6.63) in the number of training examples, while it is quadratic for the geodesic distance. This fact actually prevents the use of geodesic distances whereas the number of training elements is considerable: to give an intuition, whereas the learning of a classifier employing CBH1 takes one day, considering the geodesic distance this translates in one month.

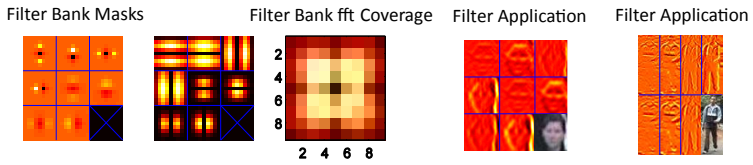


Fig. 6.21. On the left Symmetric DOOG (Difference Of Offset Gaussian) filters used to populate the feature set Φ. On the right two examples of their application on an head and a human image.

6.4.3.2 Features

In the proposed approach, from each image I ($r \times r$ pixels) is extracted a set $\Phi(I, \mathbf{x}, y)$ of dimension $r \times r \times d$ features where $d = 13$ and $\mathbf{x}, y$ are the pixel location. It is composed by:

$$\Phi(I, x, y) = \left[F_1(Y) \dots F_8(Y)\ Y\ C_b\ C_r\ G_{|\,|}(Y)\ G_O(Y)\right], \tag{6.65}$$

where $F_1(Y) \dots F_8(Y)$ is the filter bank, depicted in Fig. 6.21, consisting of scaled symmetric DOOG (Difference Of Offset Gaussian) [Dol], applied only on the luminance channel of the perceptually uniform CIELab color space. Y, C_b, and C_r are the three color channels obtained by transforming the original RGB image. $G_{|\,|}(Y)$ and $G_O(Y)$ are the gradient magnitude and orientation calculated on the Y channel map, respectively.

The covariance of the color channels are adopted, since it permits to implicitly detect hair and skin textural properties. This particularly helps in distinguishing frontal from back views (in the head orientation classification task). Moreover, the DOOG filters emphasize facial details, such as the vertical orientation of the nose, or the horizontal orientation of the mouth, if visible. Different combinations of filters are tried, like Gabor filters, the Berkeley filter bank for textons, the Laptev and Lindberg filter bank, and a set of separable steerable filters. The DOOG filters have shown to represent the best compromise among all those tested because their symmetry is an important characteristic for detection and classification, where the OIs are symmetric objects like humans and heads. The combination of these filters, codified by the covariance, is sufficiently representative to model the variability of the classes considered also in low resolution conditions. Last but not least, the DOOG filter set is compact enough to keep well bounded both the feature space dimension (avoiding the curse of dimensionality) and the computational cost of the framework.

[2] Actually, here it is not check whether $d_{\mathcal{CBH}1}$ is actually a distance in a rigorous mathematical sense. It is indeed symmetric, positive, and zero if and only if the points coincide, but one should further prove that it fulfils the triangle inequality; however, for our comparison purposes, one can safely call it, informally, distance.

6.4.4 Experiments

In this Section, the proposed approach is extensively tested for essentially achieving two goals. The first objective (Sec. 6.4.4.2) is to show how facts and intuitions of Sec. 6.4.2 can be observed on different real datasets, whose samples are located in Sym_d^+. More specifically, the sectional curvature $\kappa_{\mathbf{I}_d}$ is analysed, showing that these values are bounded in the interval $[0, -1]$ (Sec. 6.4.2.4). A statistics over the intra-dataset distances employing the Frobenius distance $d_{\mathcal{E}}$ (6.62) is also derived, the CBH1 distance $d_{\mathcal{CBH}1}$ (6.63) and the geodesic distance $d_{\mathcal{G}}$ (6.64), so as to demonstrate that, in average, $d_{\mathcal{E}} \leq d_{\mathcal{CBH}1} \leq d_{\mathcal{G}}$ (see Sec. 6.4.2.5). In addition, these experiments seem to indicate that the lower the covariance matrices dimension, the higher the curvature.

As second goal, in Sec. 6.4.4.3-6.4.4.6, a simple linear SVM is tested in classification and regression tasks, namely, head orientation classification in Sec. 6.4.4.3 and 6.4.4.4, and human orientation classification in Sec. 6.4.4.5. This is done under different operative conditions, with in total 6 datasets, each of them bringing in different issues. The proposed framework is also compared with known competitors, showing convincing performances.

6.4.4.1 Datasets

For head orientation classification, the *QMUL* is considered, the *Heads Of CoffeBreak* (HOCoffee), and the *Heads of IIT* (HIIT) [Tosa] datasets. All the datasets are partitioned into a train and test set.

The QMUL head dataset (see Fig. 6.22(d) for some examples) is formed by head images taken from the i-LIDS dataset [Off08] portraying an airport indoor scenario. It is composed by 18667 images, uniformly partitioned into 5 classes: Back (BA), Front (FR), Left (LE), Right (RI), and Background (BG). Background images contain portions of the background scene. The images are 50×50 pixels. The best performances are achieved in Sec. 6.3 in this case. The challenges of this dataset consist in scarce/non-homogeneous illumination, and quite severe occlusions.

The HOCoffee dataset (see Fig. 6.22(b)) is a novel benchmark dataset extracted from the CoffeeBreak social signal processing dataset [CBP+11], where an outdoor coffee break session during a summer school was captured, for detecting automatically social interactions. It is composed by 18117 head examples of 50×50 pixels, uniformly partitioned into 6 different classes (orientations): Back, Front, Front-Left, Front-Right, Left, and Right. The images contain a margin of 10 pixels on average, so the actual average dimension of the heads is 30×30 pixels. HOCoffee images show two main issues: the heads are captured automatically by a head detector, therefore they are often not centered in the images. In addition, there are several important occlusions.

The HIIT dataset (see Fig. 6.22(a)) has been built combining some indoor image data captured in a controlled scenario (a vision lab) and the Pointing04 [Gou], Multi-PIE [GMC+07], and QMUL [Tosa] datasets. As the previous dataset, it has 6 classes, 2000 examples each. The size of the samples is 50×50 pixels, without margin around the heads. The main characteristic of this dataset is that it has a stable background and no occlusions, so that it represents the ideal scenario where to evaluate how well a classifier can perform at a given resolution.

The QMUL and the HIIT dataset contains the images of the head of thousand of different subjects, while the HOCoffee focuses on 15 subjects taken in two different experimental sessions.

Considering the head orientation estimation, the attention is focused on two public datasets, i.e., *IDIAP* and *CAVIAR*. The IDIAP Head Pose dataset [Odo] (see Fig. 6.22(f)) comes from 8 meeting sequences of 360×288 frame resolution, where two individuals were captured while discussing about various topics in a 4-person dialogue scenario. The total number of different subjects captured is 15. They had their head orientations continuously annotated using a magnetic field location and orientation sensor tracker. The video repository has been employed for the CLEAR2007 head orientation estimation contest, following the protocol described in [BO05] (75×75 21152 samples were selected as training data and 23991 as testing data). Since the training samples are particularly biased on certain orientations, they are flipped and then a subset of 5288 images is randomly extracted, obtaining a balanced training pool. It represents a valuable benchmark set since the annotations express the pan, tilt and roll angles of the head pose.

The CAVIAR dataset [Fis] (see Fig. 6.22(e)) is a more challenging set for the estimation task due to the low resolution of the images and the presence of occlusions. The considered head samples, resized to 50×50 pixels, come from a set of sequences which have 1500 frames on average, acquired from a real surveillance camera located in a shopping centre in Lisbon. The dataset is composed by two subsets: the first is made by non-occluded head images for a total number of 21326 examples, the second consists in 366 occluded examples.

Finally, for the body orientation task, a novel dataset dubbed Human Orientation Classification (HOC) [Tosa] is introduced. Even if this task has recently attracted the attention of researchers (see for example [EG10]) no public available datasets are present in the literature (excep the ViPER dataset [GBT07], but it has a very low number of elements, limited to 632). HOC (see Fig. 6.22(c)) is derived by the ETHZ [Sch] human re-acquisition dataset representing pedestrians in different orientations and (background) conditions, captured by hand-held cameras. ETHZ is structured in three sequences for a total of 8555 images, each image 64×32 pixels containing a pedestrian. The images are manually splat into 4 orientation classes (Front, Back, Left, and Right), individuating a training and a testing partition. The dataset is complex because of the low resolution, severe illumination artefacts, occlusions and consistent scale changes. The main characteristics of all the previous presented datasets and the relative WARCO architecture instances are summarized in Tab. 6.2.

6.4.4.2 Geometrical properties of $\mathbf{Sym_d^+}$

The numerical evaluation of the curvature $\kappa_{\mathbf{I}_d}$ in correspondence of the samples of a particular dataset allows to understand how concave is the related region of Sym_d^+. In Tab. 6.3, the mean value and the standard deviation of $\kappa_{\mathbf{I}_d}$ for all the datasets are reported (note that QMUL† refers to the QMUL dataset with the background class). These values are calculated by considering each covariance matrix of WARCO as an independent sample, for all the WARCO descriptors of a single dataset. First, the mean values are all negative and almost near to 0,

Fig. 6.22. Examples of the (a) HIIT, (b) HOCoffee, (c) HOC, (d) QMUL, (e) CAVIAR, and (f) IDIAP datasets used in the experimental part. In (a), (b), (c), and (d), each row correspond to a different class. In (e) and (f), head orientation is estimated by regression. Examples are ranked from the left to the right proportionally to their degree of difficulty.

Dataset name	Dataset attributes			WARCO attributes	
	obj. of int.	♯ images	avg. obj. dim.	patches number	patch dimension
QMUL	head	16k	50×50	25	16×16
QMUL †	head	20k	50×50	25	16×16
HIIT	head	24k	50×50	25	16×16
HOCofee	head	18k	50×50	25	16×16
CAVIAR (Clean)	head	21k	50×50	25	16×16
CAVIAR (Occluded)	head	22k	50×50	25	16×16
IDIAP	head	66k	75×75	25	24×24
HOC	human	11k	62×132	40	24×24

Table 6.2. Dataset characteristics.

evidencing the presence quite flat regions. Furthermore, larger patches seem to lie in flatter regions, and this assumption will be validated heuristically, in a more exhaustive fashion, later in the section.

Looking at the different distance statistics in the same table, related to the Frobenius distance $d_{\mathcal{E}}$ (6.62), the CBH1 distance $d_{\mathcal{CBH}1}$ (6.63), and the geodesic distances $d_{\mathcal{G}}$ (6.64), one can note that for the mean values the inequality $d_{\mathcal{E}} \leq d_{\mathcal{CBH}1} \leq d_{\mathcal{G}}$ holds systematically; on the contrary, the standard deviation does not present the same behaviour and its values look similar.

Dataset name	$\kappa_{\mathbf{I}_d}$		$d_{\mathcal{E}}$		$d_{\mathcal{CBH}1}$		$d_{\mathcal{G}}$	
	mean	standard dev.	mean	standard dev.	mean	standard dev.	mean	standard dev.
QMUL	−0.035	0.017	7.78	2.72	8.21	2.70	8.78	2.62
QMUL †	−0.038	0.020	8.65	3.02	9.13	3.03	9.65	2.89
HIIT	−0.031	0.018	7.02	3.74	7.41	3.77	8.02	3.86
HOCofee	−0.035	.0.15	6.40	2.57	8.37	3.34	8.88	3.24
CAVIAR (Clean)	−0.041	0.021	8.59	2.24	9.16	2.57	9.73	2.50
CAVIAR (Occluded)	−0.043	0.026	8.12	2.72	8.88	2.88	9.12	2.73
IDIAP	−0.014	0.006	4.79	1.81	5.01	1.83	5.34	1.83
HOC	−0.024	0.014	7.67	3.36k	7.99	3.34	8.41	3.25

Table 6.3. Curvature analysis and distance comparison of different datasets. $\kappa_{\mathbf{I}_d}$, $d_{\mathcal{E}}$, $d_{\mathcal{CBH}1}$, and $d_{\mathcal{G}}$ are compared on the same covariance representation (see Eq. (6.65)). See Sec. 6.4.4.2 for details.

6.4.4.3 Head Orientation Classification

QMUL Head dataset. WARCO classifier is tested adopting both the Frobenious and the CBH distances, against the template-based discriminative approach presented in [OGX09] and the ARCO LogitBoost-based strategy (see Sec. 6.3), the latter being the current best approach. To reproduce the former method, the image features provided by the dataset authors are considered and the same experimental protocol is followed. The same polynomial SVM classifier as in [OGX09] is also used. The confusion matrices are reported in Fig. 6.23, considering 4 and 5 (4 orientations plus the background) classes. WARCO with CBH1 distances get the highest average classification scores. One should also pay attention to Fig. 6.23(h), where the accuracy in classifying the background class rises of about 10% with respect to the previous state-of-the-art results depicted in Fig. 6.23(f). This gap is due to the CBH distance: actually, background samples are located in zones with higher curvature (validated experimentally), far from $\mathbf{I}_d$, so that the contribution given by the CBH expansion becomes critical in better capturing the local geometry.

HOCoffee dataset.

In this case, one has 6 orientations. In Fig. 6.24(e), the qualitative performances, and in Fig. 6.24(c) and (d), the quantitative performances are reported considering both Frobenious (FROB) $d_{\mathcal{E}}$ distance (6.62) and the CBH1 $d_{\mathcal{CBH}1}$ distance.

HIIT dataset.

As one can note in Fig. 6.24(a) and (b), the performance of our framework are rather high, in fact, using $d_{\mathcal{CBH}1}$ to measure the distance among covariance matrices the average accuracy is 97%. This means that the proposed classifier manages easily low resolution head images classifying the orientation precisely.

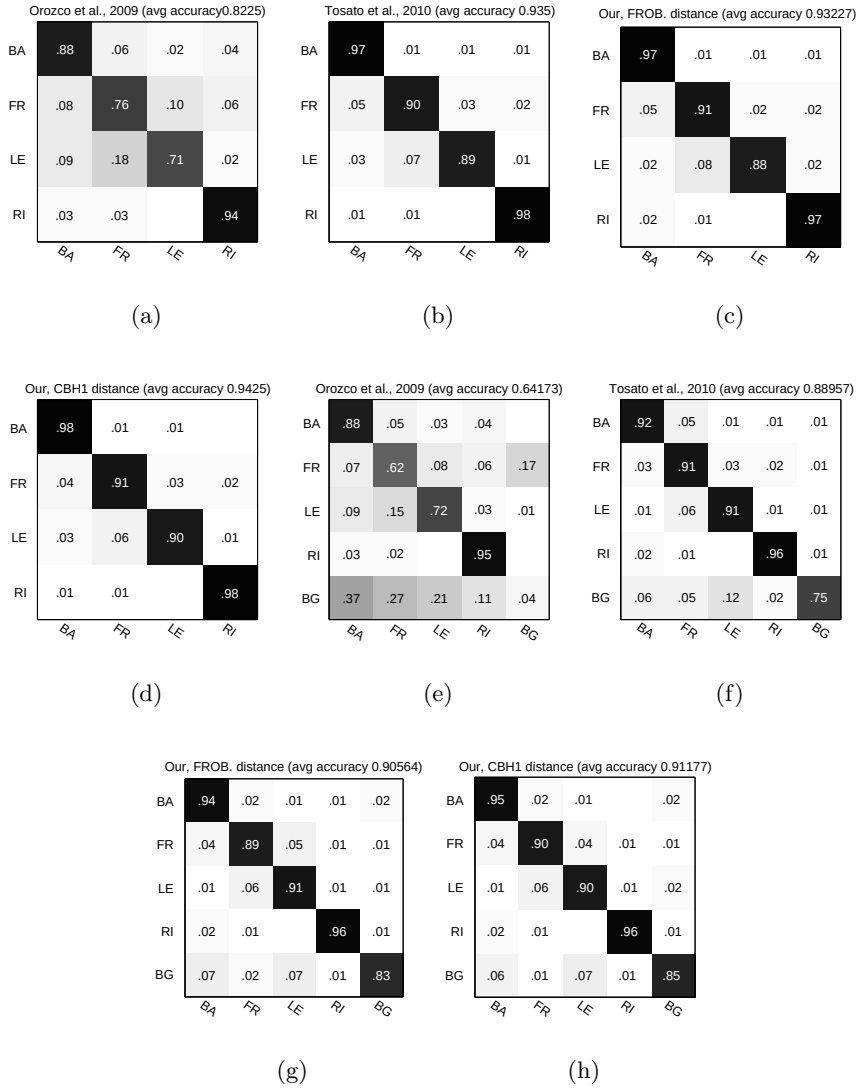
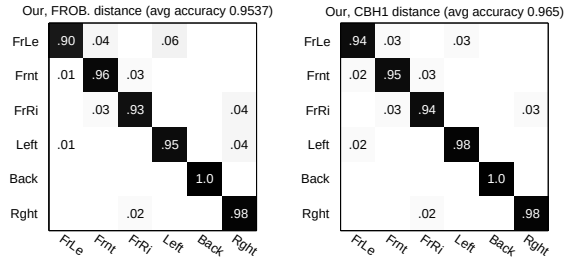


Fig. 6.23. Examples and statistics of the 4 and 5-class original dataset taken by Orozco et al. [OGX09]. (a) and (e): the original results by Orozco et al. approach [OGX09]. (b) and (f): the Tosato et al. approach described in Sec. 6.3. (c), (d), (g), and (h): , the proposed approach.

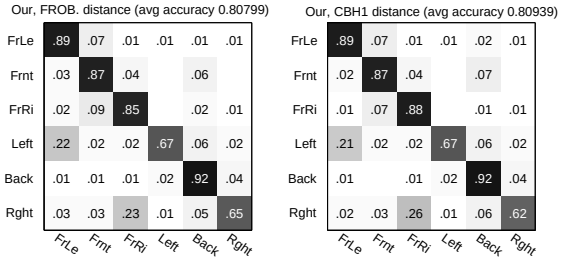
6.4.4.4 Head Orientation Estimation by Regression.

In this context, the SVM classifier is replaced with a linear SVM regressor [CL].

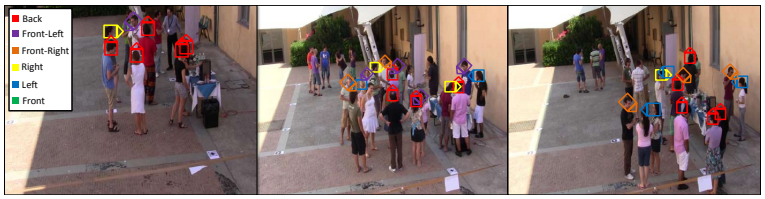
IDIAP Head Pose. The head orientation evaluation protocol is taken from [BO05]: in each one of the 8 meetings of the test set, one has 1 minute of recording for the testing, for a total of 1500 test samples. The three error measures suggested



(a) (b)



(c) (d)



(e)

Fig. 6.24. Confusion matrices on the (a) and (b) HIIT, and (c) and (d) CoffeeBreak head orienation datasets [Tosa]. (e) shows a qualitative result on the CoffeeBreak dataset.

	pan			tilt			roll		
	mean	std	med	mean	std	med	mean	std	med
Hist+Correlation	16.2	13.6	13.1	22.4	15.0	19.1	15.1	12.0	12.5
Correlation+Shape	19.0	17.4	14.2	26.4	17.5	21.5	16.1	12.7	13.4
Texture	13.6	14.9	8.3	17.6	13.8	12.8	11.5	10.3	12.9
Texture+Color	**8.7**	**9.1**	**6.2**	19.1	15.4	14.0	9.7	7.1	8.6
Our, FROB. distance	10.90	10.75	7.87	4.81	5.98	2.93	4.65	4.22	3.80
Our, CBH1 distance	10.30	10.61	7.13	**4.46**	**5.26**	**2.54**	**4.33**	**3.84**	**3.33**

Table 6.4. Pan, tilt and roll error statistics over evaluation data of IDIAP dataset. The first 4 methods are taken from [BO05].

by the protocol are adopted, which are the absolute differences with the ground-truth pan, tilt and roll angles. Table 6.4 summarizes our results considering the methods that participated to the CLEAR2007 challenge [BO05], that defined the best performances on this dataset. As one can observe, good results concerning the pan are reached, while the best scores with the tilt and roll angles are defined.

CAVIAR. The best competitor for this dataset are considered, which is the

	pan		
	mean	std	med
Robertson & Reid [RR06]	76.4	55.8	70.1
WARCO, Clean, FROB. distance	22.65	18.44	17.09
WARCO, Clean, CBH1 distance	22.21	18.38	16.90
WARCO, Occluded, FROB. distance	36.90	25.23	31.73
WARCO, Occluded, CBH1 distance	35.26	24.58	30.70

Table 6.5. Pan error statistics over evaluation data of CAVIAR dataset both for non-occlluded and occluded cases.

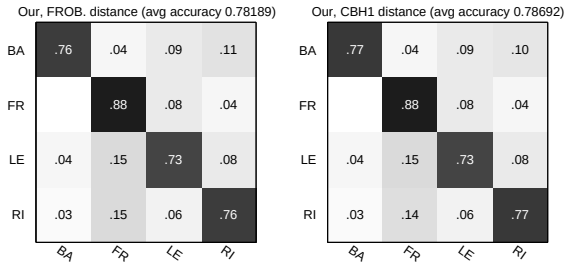
method presented in [RR06]. Unfortunately, it is difficult to produce a fair comparison. In this paper [RR06], ground truth annotations are made by the authors, which unfortunately are not compatible with that provided together with the dataset. In practice, they represent a quantized version of the original annotations. Employing the original annotations, two datasets are individuated, one formed by non-occluded samples, the other with occlusions, and the pan angle are estimated on both sets. Results are shown in Table 6.5, where, as in [RR06], the mean, the standard deviation and the median of the errors are reported. Two main considerations pop out. The first one is that the proposed approach gets lower errors than [RR06]. Apart from the different methodologies in getting ground truth data, that should make the task of [RR06] easier than ours, WARCO is noticeably more accurate. The second observation is that the errors of WARCO in the occluded cases are not dramatically higher than the un-occluded cases, and this is due to the nature of WARCO, i.e., an ensemble of local classifiers.

6.4.4.5 Human Orientation Classification dataset.

In this case, WARCO is computed on 40 overlapped patches of 24×24 pixels. In Fig. 6.25 one can see the accuracy result achieved by our algorithm. Despite the heavy occlusions and the bad illumination conditions, the average accuracy reaches 79%. It is worth noting how the Front and the Back classes are nicely separated: this is an impressive results, since here the most noticeable difference between the two classes lies in the head portion, which is relatively small.

6.4.4.6 Scale issues.

Here, I stress the capability of WARCO of working at low resolution, and the relation between patch dimensions and manifold curvature $\kappa_{\mathbf{I}_d}$ is explored. Two additional experimental sessions are produced, where the image dimensions of



(a) (b)

Fig. 6.25. Confusion matrices showing the performances between the WARCO method using (a) the Frobenius distance ($d_{\mathcal{E}}$) and (b) the CBH1 distance ($d_{\mathcal{CBH}1}$).

each dataset is reduced by a factor of 0.75 and 0.5. Consequently, the same factor of reduction is applied to the architecture of WARCO. In Fig. 6.26, the results concerning the classification are reported, and in Fig. 6.27 the results for the regression task are shown. As one can note, the smaller the size of the object, the higher the curvature. Furthermore, it is valuable to observe how the CBH1 distance-based framework behaves with respect to the Frobenious distance-based technique at the different resolutions: the lower the resolution, the bigger the gap between CBH1 and the Frobenious-based strategy. Once again, this demonstrates that the contribution of CBH1 is in general more helpful in highly curved manifold regions.

6.5 Fast and Robust Inference with WARCO

In this Section, instead of employing the linear SVM learning framework as in Sec. 6.4, Random Forest (RF) are adopted (see Sec. 3.3.2) to train the WARCO (see Sec. 6.4 for details) patch models because they have several interesting properties.

In fact, RF can actually be trained on large datasets with a low computational cost and without being affected by significant overfitting: this allows to train several classifier (or regressor) instances, one per patch, rapidly. Moreover, RF is very efficient during the testing phase, since labelling an example against a tree has logarithmic cost in the number of leaves, and can also tolerate a certain amount of noise and errors in the training data labels.

The goal is to understand if, using a much efficient, robust to noisy data, and intrinsically nonlinear learning framework as RF, it is possible to obtain good performances with WARCO. Therefore, I find out a model, FWARCO (Fast WARCO), that exploits RF to use the power of WARCO efficiently.

Due to the good properties of RF, to obtain a high robustness to the false positives during the testing phase, it is important to use large training sets of background examples. In this situation, the training problem is highly unbalanced because of huge cardinality of background ROIs. In the same vein of [FGMR10], a

Dataset	Obj. Size	$\kappa_{\mathbf{I}_d}$	$d_{\mathcal{E}}$ avg acc.	$d_{\mathcal{CBH}1}$ avg acc.
QMUL	25×25	-0.0509	78%	80%
	38×38	-0.0448	89%	90%
	50×50	-0.0361	91%	92%

(a)

Dataset	Obj. Size	$\kappa_{\mathbf{I}_d}$	$d_{\mathcal{E}}$ avg acc.	$d_{\mathcal{CBH}1}$ avg acc.
QMUL †	25×25	-0.0571	74%	76%
	38×38	-0.0470	86%	87%
	50×50	-0.0345	90%	91%

(b)

Dataset	Obj. Size	$\kappa_{\mathbf{I}_d}$	$d_{\mathcal{E}}$ avg acc.	$d_{\mathcal{CBH}1}$ avg acc.
HIIT	25×25	-0.0571	88%	90%
	38×38	-0.0571	95%	96%
	50×50	-0.0571	96%	96%

(c)

Dataset	Obj. Size	$\kappa_{\mathbf{I}_d}$	$d_{\mathcal{E}}$ avg acc.	$d_{\mathcal{CBH}1}$ avg acc.
HOCoffe	25×25	-0.607	62%	66%
	38×38	-0.0430	78%	80%
	50×50	-0.0345	80%	80%

(d)

Dataset	Obj. Size	$\kappa_{\mathbf{I}_d}$	$d_{\mathcal{E}}$ avg acc.	$d_{\mathcal{CBH}1}$ avg acc.
HOC	66×31	-0.0320	71%	73%
	99×47	-0.0230	77%	78%
	132×62	-0.0192	78%	78%

(e)

Fig. 6.26. Comparative study of the performances of the proposed statistical classification framework.

hard negative mining strategy is designed for RF here. Also in this case it is possible to prove that data-mining methods can be made to converge to the optimal model defined in terms of the entire training set. Exploiting the proposed hard mining, it is shown how to increase the robustness of WARCO and the experimental evidences demonstrate this fact.

The rest of the Section is organized as follows. In Sec. 6.5.1, the classification model able to deal with both head and body orientation classification is described and in Sec. 6.5.2 WARCO and FWARCO are compared experimentally.

6.5.1 The Approach

The set of patches which composes FWARCO is denoted with $\Phi = \{\phi_i\}_{i=1,\ldots,N_P}$ (the same as in WARCO), where N_P is the number of image patches, described by a set of $d \times d$ covariance matrices. Concisely, the idea below the proposed framework is that each patch classifier votes for a class, and the final classification

			Avg Pan Err.	
Dataset	Obj. Size	$\kappa_{\mathbf{I}_d}$	$d_{\mathcal{E}}$	$d_{\mathcal{CBH}1}$
CAVIAR (Clean)	25×25	-0.0437	27.15	25.63
	38×38	-0.0426	22.65	21.58
	50×50	-0.0415	19.74	19.73

(a)

			Avg Pan Err.	
Dataset	Obj. Size	$\kappa_{\mathbf{I}_d}$	$d_{\mathcal{E}}$	$d_{\mathcal{CBH}1}$
CAVIAR (Occluded)	25×25	-0.045	41.00	38.00
	38×38	-0.044	37.00	36.33
	50×50	-0.043	36.90	35.26

(b)

			Avg Pan Err.		Avg Tilt Err.		Avg Roll Err.	
Dataset	Obj. Size	$\kappa_{\mathbf{I}_d}$	$d_{\mathcal{E}}$	$d_{\mathcal{CBH}1}$	$d_{\mathcal{E}}$	$d_{\mathcal{CBH}1}$	$d_{\mathcal{E}}$	$d_{\mathcal{CBH}1}$
IDIAP	38×38	-0.0293	16.18	16.07	6.67	6.47	5.02	4.97
	56×56	-0.0175	12.35	12.03	5.18	5.01	4.94	4.82
	75×75	-0.0143	10.90	10.30	4.81	4.46	4.65	4.33

(c)

Fig. 6.27. Comparative study of the performances of the proposed statistical regression framework.

is the weighted summation of the votes. The same model is instantiated also for the regression task to obtain a continuous output. Since the goal is to infer the body or head orientation, the final (output) orientation is the median orientation given by the contribution of each model patch. As anticipated, RF is the algorithm adopted to learn the model for each single patch, which is the same both for classification and regression. However, the same statistical model described in Sec. 6.4.3 is used here to assign a label.

6.5.1.1 Training Models

Starting from the Alg. 6, introduced in Sec. 3.5.1, its adaptation for RF is described. Considering one WARCO patch, I refer to its training set by

$$\{\mathbf{X}^i, y^i\}_{i=1,\dots,S},$$

where $\mathbf{X}^i$ are the covariance matrices of an image patch. It is worth noting that the notation, in this case, is simplified omitting the patch index.

I recall briefly the concept under RF (see Sec. 3.3.2 for more details). It is a combination of decision trees such that each tree depends on the values of a random vector, sampled independently and with the same distribution for all the trees in the forest. Trees are grown randomly using binary partitioning. In particular, randomness is injected by growing each tree on a different random sub-sample of the training data into the splitting process so that a small subset of randomly selected features is used for the splitting decision. For this technique, four parameters must be adopted: (1) for each node, the feature to split a node is selected

among a random subset of all the d_v features; the number of candidate feature is fixed to $\sqrt{d_v}$; (2) to guarantee good generalization performances of the classifier the number of samples per leaf is fixed to at least τ; (3) each tree is trained on a randomly drawn bootstrap sub-sample of the data, and here it is fixed using approximately 2/3 of the examples; (4) the number of trees is fixed to $T = 100$ to reduce the amount of memory necessary to instantiate the classifier, since the implementation used [Jai] is not optimized. This setting holds for all the experiments reported in Sec. 6.5.2. To maximize the speed of FWARCO, the Frobenius norm on Sym_d (Eq. 6.62) is used to measure the distance between covariance matrices.

Algorithm 13: Random Forests on Sym_d

Data: $\mathcal{D} = (\mathbf{X}^1, y^1), \ldots, (\mathbf{X}^S, y^S)$ with $\mathbf{X}^i \in Sym_d^+$ and $\mathbf{Y}^i \in \{1, \ldots, L\}$, T the number of trees, τ the minimum number of examples per tree, $m = \sqrt{d(d+1)/2}$ the feature to split a node.
Result: The ensemble of classifiers $\{g_1, \ldots, g_T\}$.
begin
  Normalize the covariance matrices using Eq. (5.21);
  Map the data points to the tangent space $T_{\mathbf{I}_d}$, by $\tilde{X}^i = \log_{\mathbf{I}_d} \mathbf{X}^i$ (Eq. (6.25));
  Vectorize $\tilde{X}^i$ as $\mathbf{x}^i = \mathrm{vec}(\tilde{X}^i)$ (Eq. (2.1));
  for $t = 1, 2, \ldots, T$ **do**
    Randomly sample the training data $\mathcal{D}^i \subset \mathcal{D}$ with replacement to produce $\mathcal{D}_i$;
    Create a root node, N^i, containing $\mathcal{D}^i$;
    while $|N^i| > \tau$ **do**
      Randomly select m of the possible splitting features in N^i;
      Grow the tree g_t, splitting current node with the best variable among the m selected;

6.5.1.2 Strategies for robust detection

The learning framework proposed in the previous section gives an effective solution to the object classification problem, but not for detection problems. Actually, a very large number of background patches can be computed from a single image in the latter case. A different approach from the well-known class of bootstrapping methods is proposed, where multiple classifiers are learned during the training phase to enhance the final classifier robustness.

Recently, in [FGMR10], a data-mining strategy to prune easy-to-classify background examples is introduced. This technique has been designed for margin-based classifiers like SVM, but since RF is adopted in this Section, it is necessary to choose a slightly different strategy to tackle this problem using the same basic idea as the one used for SVM. In any case, it is possible to prove that data mining methods converge to the optimal model defined in terms of the entire training set.

The main advantage in using this data pruning, instead of the cascade approach, is related to the classification efficiency.

Before showing the data mining procedure for RF, it is necessary to define hard and easy background examples of a training set $\mathcal{B} \subset \mathcal{S} \setminus \mathcal{V}$ as follows:

$$\mathcal{H} \subseteq \mathcal{B} = \left\{ \mathbf{x} \in T_{\mathbf{I}_d} Sym_d^+ \,|\, \mathrm{P}(BG|\mathbf{x}) < 1/2 \right\}, \tag{6.66}$$

are the hard examples and, similarly, the easy examples $\mathcal{E}$ are defined. $\mathrm{P}(BG|\mathbf{x})$ is the posterior probability, given by Eq. (5.24), of a background example to belong to the background class. Given a large background training set $\mathcal{B}$, the goal is to find a smaller set of examples $\mathcal{B}^* \subseteq \mathcal{B}$ such that $f_n(\mathcal{B}^*) = f_n(\mathcal{B})$.

Let $\mathcal{S}_1$ ($t = 1$) an initial subset of m randomly selected examples for the BG class joined to all the examples of the other (foreground) classes.

This method iteratively trains a model and updates the dataset in the following way:

1) train the model f_n using RF and $\mathcal{S}_t$ as training set.
2) If $\mathrm{P}_{\Delta\mathrm{err}}(\mathcal{B}) = 0$ stop and return it.
3) Let $\mathcal{B}' = \mathcal{B} \setminus \mathcal{B}^*$, where $\mathcal{B}^* \subset \mathcal{S}_t$ contains all the $\mathcal{S}_t$ background examples.
4) Compute $\mathcal{H} \subseteq \mathcal{B}'$ (Eq. (6.66)) and $\mathcal{E} \subseteq \mathcal{B}$.
5) Compute $\mathrm{P_t}(\mathcal{B}) = \frac{\sum_i 1\{f_n(\mathcal{B}) \neq BG\}}{|\mathcal{B}|}$ and $\mathrm{P}_{\Delta\mathrm{err}}(\mathcal{B}) = |\,\mathrm{P_t}(\mathcal{B}) - \mathrm{P_{t\text{-}1}}(\mathcal{B})|$, where $1\{\cdot\}$ is an indicator function.
6) Update $\mathcal{S}_t$ by adding a subset of m examples of $\mathcal{H}$ and removing $\mathcal{E}$. Go to 2.

In this procedure, all foreground examples are considered for each iteration, where easy background examples are removed from the set of possible candidates and new hard background examples are added to the current training set, pruning all the easy negatives. Since the goal is to minimize the training set dimension, a small number m of hard background examples is added to $\mathcal{B}_n^*$. At the first iteration, the number of background examples is exactly m and grows when the number of iterations increases. However, I observed from the experimental trials that the final number of background examples naturally stops growing in few iterations. In particular, it stops when the number of negatives is similar to the number of positives.

It is also possible to build an approximate version of the previous hard mining procedure in order to speed up its convergence in presence of noisy and mis-labelled training data. Denoting with $\mathrm{P_{err}}$ the normalized fraction of mis-labelled examples, the exit condition can be relaxed from $\mathrm{P_{err}}(B_n) = 0$ to $\mathrm{P_{err}}(B_n) \leq \epsilon$ (ϵ is fixed to 0.01 in these cases). This approximated method guarantees both good classification performances and a fast convergence of the algorithm. It is possible to show, with the following lemma stating, that when the method stops one has found $f_n(B_n^*) = f_n(B_n)$.

Lemma 6.1. *Let $B_n^* \subseteq B_n$. If $H \subseteq B_n^*$ then $f_n(B_n^*) = f_n(B_n)$ in probability.*

Proof. $B_n^* \subseteq B_n$ means that $\mathrm{P_{err}}(B_n^*) \leq \mathrm{P_{err}}(B_n)$. Since $H \subseteq B_n^*$, all the examples in $B_n \setminus B_n^*$ are classified correctly by $f_n(B_n^*)$. For this reason $\mathrm{P}_{\Delta\mathrm{err}}(B_n^*) = 0$. One concludes that $f_n(B_n^*) = f_n(B_n)$ in probability.

The next theorem shows that this method always terminates. The idea is that the dimension of B_n^* grows at each iteration and is bounded by B_n.

Theorem 1 *Let $B_n^* \subseteq B_n$, hence the hard mining procedure terminates in a finite number of iterations.*

Proof. One can denote $B_{n_t}^* \subset S_t$ as the set of background examples at iteration t and f_{n_t} as the classifier learned at t. Since $B_{n_t}^* \subseteq B_{n_{t+1}}^*$, it implies $f_{n_{t+1}} \geq f_{n_t}$. At the end of the process, B_n^* contains all the hard negative examples from B_n. This implies that $f_n(B_n^*) = f_n(B_n)$ in probability. During the process $H \subseteq B_n'$ at $t+1$ contains at least one example $(\mathbf{x}, y)$ mis-classified by f_{n_t}. Since $B_{n_t}^* \subseteq B_{n_{t+1}}^*$ one has $f_{n_{t+1}} \geq f_{n_t}$ in probability. If $B_{n_t}^* \neq B_{n_{t+1}}^*$, then $f_{n_{t+1}} > f_{n_t}$ due to $(\mathbf{x}, y)$. If $B_{n_t}^* = B_{n_{t+1}}^*$, then $f_{n_{t+1}} = f_{n_t}$. Therefore $f_{n_{t+1}} \geq f_{n_t}$, since there are finitely many B_n^* that can be built in a finite number of iterations.

6.5.2 Experimental Results

Sec. 6.5.2.1,6.5.2.2, and 6.5.2.3 presents different applications of FWARCO. In particular head orientation classification in Sec. 6.4.4.3 and Sec. 6.4.4.4, and human orientation classification in Sec. 6.4.4.5. The results of FWARCO with the best ones of WARCO are compared, previously presented in Sec. 6.4.4, using the same datasets and the same features. The goal is to understand if, using a much efficient, robust to noisy data and intrinsically nonlinear learning framework as RF, it is possible to obtain good performances with WARCO. In the following experiments, to maximize the speed of FWARCO, the Frobenius norm on Sym_d (Eq. 6.62) is used to measure the distance between covariance matrices. The results given by ARCO (see Sec. 6.3) are omitted because of the clear superiority of WARCO.

As experimental data, the most interesting dataset used in Sec. 6.4) are considered. They are the QMUL Head Pose [Tosa], the IDIAP Head Pose [Odo] and HOC [Sch]. Then a novel dataset is introduced. It is probably the most challenging for the human orientation classification task which is extracted from the ViPER dataset [GBT].

6.5.2.1 Head Orientation Classification

QMUL Head Pose. For the head orientation classification task, FWARCO and WARCO are compared, described in Sec. 6.4, using the same feature set. The result of the comparison, in terms of confusion matrix, is reported in Fig. 6.28. The average rate is 89% for FWARCO, against 91% of WARCO, therefore FWARCO loses only a 2% in accuracy, but it increase dramatically the performance in terms of efficiency during the testing phase. This permits to use a much greater number of background examples and to exploit the procedure described in Sec. 6.5.1.2, it is possible to increase the robustness against the background almost to 99% of accuracy for the background class, only losing 1% of the accuracy in average.

(a) (b)

Fig. 6.28. (a) The confusion matrices for the head orientation classification with WARCO 6.4. (b) FWARCO results.

6.5.2.2 Head Orientation Estimation by Regression

IDIAP Head Pose. For the estimation task, FWARCO is tested on the IDIAP Head Pose dataset, which is used for the CLEAR 2007 head orientation estimation challenge. Tab. 6.6 summarizes the results of this system. As for the previous case,

	pan			tilt			roll		
	mean	std	med	mean	std	med	mean	std	med
WARCO (Sec. 6.4)	10.30	10.61	7.13	4.46	5.26	2.54	4.33	3.84	3.33
FWARCO	13.9	12.10	10.52	5.44	4.93	2.01	4.14	3.56	3.01

Table 6.6. Pan, tilt and roll error statistics over evaluation data for the WARCO 6.4 and FWARCO.

the same experimental setting used for WARCO is reproduced and also in this case the same feature set is adopted. RF ability to model non-linear boundaries and its ability to manage noisy data leads for the tilt and roll, if compared to WARCO.

More detailed results of FWARCO on this database are shown in Fig. 6.29, where it is compared with some 4 methods (M1-4) taken from [BO05]. The results shown in this figure report how the system works for each single meeting.

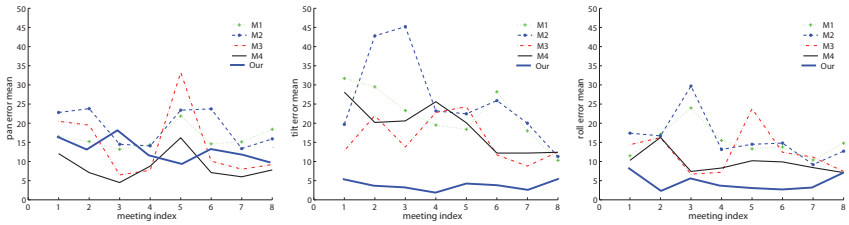


Fig. 6.29. Mean of pan, tilt and roll orientation classification errors for individual meeting evaluation data of the IDIAP dataset.

6.5.2.3 Human Orientation Classification

As in Sec. 6.4.4.5, the HOC dataset [Tosa] is used. Then, another public dataset for the human orientation classification task introduced. It is derived from the well known ViPER [GBT] dataset. It represents pedestrians in different orientations and (background) conditions. I named this dataset ViPER human orientation dataset [Tosa]. However, both datasets are challenging because pedestrian images are represented at low resolution and they are taken from video surveillance scenarios where many illumination changes and occlusions occur.

HOC. In Fig. 6.30 the results of the comparison between WARCO and FWARCO are reported. FWARCO achieves the best result in accuracy thanks to its intrinsic non-linear separation ability and its robustness to noise. Despite the heavy occlusions and the bad illumination conditions, the average accuracy reaches 81%.

Fig. 6.30. Results on the HOC dataset. In (a) the confusion matrix associated with WARCO (see Sec. 6.4) and in (b) the FWARCO one.

ViPER Human Orientation. This dataset contains two camera views of 632 pedestrians. Each pair contains images of the same pedestrian taken from different cameras, under different viewpoints, orientations and illumination conditions. All images are normalized to 128×48 pixels. Most of the examples contains a viewpoint change of 90 degrees. Since the target is human orientation classification, the images of the two views are joined. Then, all the images are reflected vertically and small translations are performed to build a dataset of 8969 pedestrian images. In order to build a balanced training set, about 1500 images per class are randomly sampled and the testing set is composed by the remaining images. According to [EG10], the experiments are performed considering different numbers of classes: three in the first case, four in the second one. This because front and back orientation classes in this task (considering only static images) depend almost exclusively from the head orientations. So, it is not possible to build a reliable model and therefore the front/back classes in this case are heavily overlapped. it is worth noting that this fact does not affect the goodness of the results on the other datasets, because ViPER is the most challenging one.

In Fig. 6.31 some ViPER example are shown and the confusion matrices for the 3- and 4-class classification experiments are reported in Fig. 6.32. In this

Fig. 6.31. Some examples of pedestrians in four orientations taken from the ViPER human orientation dataset [Tosa].

case, the results obtained with WARCO and its linear SVM learning framework are omitted because it perform poorly with ViPER data. This because the linear model is too simple to obtain good performance on this dataset. FWARCO gives

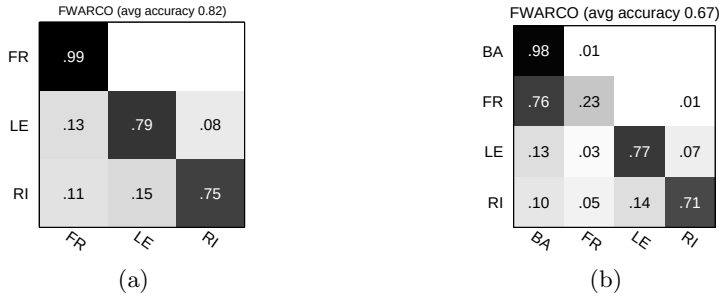


Fig. 6.32. In (a) and (b) the confusion matrices for the human orientation classification in the 3- and 4-class cases (from the left to the right, respectively), using FWARCO.

an average accuracy of 82% on this dataset which is a very good result considering the complexity of the data. Adding the fourth class, the performances obviously decrease to 67% as for the comments discussed above.

6.6 Head Orientation Classification for Social Interactions

The automatic recognition of human activities in video recordings is undoubtedly one of the main challenges for a surveillance system. This is usually accomplished using a serial architecture built upon an array of techniques aimed to extract low-level information including for example, foreground/background segmentation [BJE+08] and object tracking [FV06]. After these early processing stages, high-level analysis methods aim at detect atomic actions (e.g., gestures) as well as complex activities (i.e. spatio-temporal structures composed of atomic actions) [CRCZ05], possibly exploiting ontologies for ensuring interoperability across

different platforms and semantic descriptions understandable to human operators [FNHB05].

However, these technologies seem to forget that, for human beings, physical and social space are tightly intertwined and no intelligent monitoring is possible without taking account of social aspects associated with behaviours. This is especially regrettable when other domains, e.g. Affective Computing (AC) [Pic00] or Social Signal Processing (SSP) [VPB09], pay significant attention to social, affective and emotional aspects of human behaviour. In particular, Social Signal Processing aims at developing theories and algorithms that codify how human beings behave while involved in social interactions, putting together perspectives from sociology, psychology, and computer science [Pen07, VPB09, PPN09].

The main tools for the analysis are the social signals [VPB09], i.e. temporal co-occurrences of social cues [AR92], that can be basically defined as a set of temporally sequenced changes in neuromuscular, neurocognitive, and neurophysiological activity. Social cues are organized into five categories that are heterogeneous, multimodal aspects of a social interplay [VPB09]: 1) *physical appearance*, 2) *gesture and posture*, 3) *face and eyes behaviour*, 4) *vocal behaviour*, and 5) *space and environment*.

This Section concentrates on the Visual Focus Of Attention (VFOA) cue [SFYW99, LKTT07, SBOGP08], that belongs to the third category and is a very important aspect of non-verbal communication, taking also into account the fifth category, usually disregarded by social signalling studies [CVV10]. The VFOA indicates where and what a person is looking at and it is mainly determined by head pose and eye gaze estimation. In cases where the scale of the scene does not allow to capture the eye gaze directly, viewing direction can be reasonably approximated by simply measuring the head pose; this assumption has been exploited in several approaches dealing with a meeting scenario [SFYW99, SYW02, VS08] or in a smart environment [SBOGP08, LBC$^+$09].

Following this claim, and considering a general, unrestricted scenario, where people can enter, leave, and move freely, VFOA is approximated as the *Subjective View Frustum* (SVF), first proposed in [FBMC09]. This feature represents the three-dimensional (3D) visual field of a human subject in the scene. According to biological evidence [PZ79], the SVF can be modelled as a 3D polyhedron delimiting the portion of the scene which the subject is looking at (see Figure 6.33).

Employing SVF in conjunction with cues of the *space and environment* category allows to detect signals of the possible people's interest, with respect to both the physical environment [FBMC09] and the other participants acting in the scene. More specifically, a method to statistically infer if a participant is involved in an interactional exchange is proposed. In accordance with cognitive and social signalling studies, it is highly probable that the interaction takes place when two people are closer than 2 meters [VPB09], and looking at each other [WFDJ94, LWB00, JWVG03]. This condition can be reliably inferred by the position and orientation of the SVFs of the people involved. This information can then be gathered in a *Inter-Relation Pattern Matrix* (IRPM), that encodes the social exchanges occurred between all the participants.

Detecting social relations among people may be useful to instantiate a more robust definition of group in surveillance applications. Actually, in the last few years, several applications focused on the group modelling have been proposed [MJD+00] and re-identification [ZGX09]; in the first application a group is defined following physically-driven proximity principles, while in the re-identification groups are assumed to be detected by an external algorithm.

More broadly, the proposal is a step forward automatic inference and analysis of social interactions in general: it is alternative to the paradigm of wearable computing [Pen00, CP02], or smart rooms [WSB+03]. In the typical non-cooperative video surveillance context or when a huge amount of data is required, wearable devices are not usable. Moreover, the usage of non-invasive technology makes people more prone to act normally.

Considering the literature (except for [FBMC09]), the "subjective" point of view for automated surveillance systems has been taken into account in [BR09], that draw inspiration from [RR06], therefore representing the most similar approach in the literature to the proposed one. In that paper, the goal is to address the head orientation of low-resolution pedestrians to infer regions of interests in the scene. However, while [BR09] modelled the gaze orientation in a continuous way can restrict to a fixed number of orientations (= 4); in addition, in [BR09], interaction analysis was absent, and the subjective point of view was functional solely on the estimation of interest maps of the scene. This last point is the most important, distinctive aspect of the proposed work.

The works of [OYTM06] and [HJB+08] are also close to the proposed one as they estimate a sort of focus of attention of single individuals. They are also different because they consider a meeting scenario that is usually more constrained than a surveillance one, and can be monitored with higher accuracy. In [OYTM06], the gaze pose in high-resolution images is estimated to infer inter-personal relations. Due to the low resolution one can prefer to perform head pose estimation because eye gaze is very hard. This idea is also followed by [HJB+08]. However, they suppose that the VFOA of each person is constrained: a person can only look at another person. This assumption could be invalid in a surveillance scenario, where people can wander around freely, look at other objects in the environment, be distracted by external events during a conversation in a group etc. For this reasons, one may left unconstrained the head pose estimation.

Summarizing, the proposed framework provides two novel contributions. First, we propose a more accurate estimation of the Subjective View Frustum: in [FBMC09], head orientation is estimated by person walking trajectory. This is reasonable when he/she is moving in the scene, but it is not valid in general. A more reliable head orientation classification is introduced here, employing a multi-class boosting algorithm, operating on covariance features [TPM08]. Second, the Inter-Relation Pattern Matrix is introduced, to inferring social interactions among people in a crowded, general scenario. This work not only fills a gap in the state of the art of SSP aimed at understanding social interactions, but also represents a novel research opportunity, alternative to the scenarios considered so far in socially-aware technologies, where automatic analysis techniques for the spatial organization of social encounters are taken into account.

The rest of the Section is organized as follows. In Sec. 6.6.1, the main techniques for estimating the VFOA in absence of gaze information and the methods for head pose estimation are reviewed. In Sec. 6.6.2, the building process of the SVF estimation method is described, sketching all the involved processing steps. In Sec. 6.6.6, the Inter-Relation Pattern Matrix description is reported. In Sec. 6.6.7, experiments on home-made and public datasets are illustrated.

6.6.1 State of the art

A person's VFOA is determined by his eye gaze. Since objects are foveated for visual acuity, gaze direction generally provides more precise information than other bodily cues regarding the spatial localization of one's attentional focus. A detailed overview of gaze-based VFOA classification in meeting scenarios is presented in [BO06]. However, measuring the VFOA by using eye gaze is often difficult or impossible: either the movement of the subject is constrained or high-resolution images of the eyes are required, which may not be feasible [MOZ02, SSdVL03], and several approximations are considered in many cases. For example, in [SFYW99], it is claimed that the VFOA can be reasonably inferred by head pose in many cases. Following the same assumption, in [SBOGP08] pan and tilt parameters of the head are estimated, and the VFOA is represented as a vector normal to the person's face. It is employed to infer whether a walking person focuses on an advertisement located on a vertical glass or not. Since the situation is very constrained, this proposed VFOA model works quite well; anyway, as observed by the authors themselves, a more complex model, that considers camera position, people's position and scene structure, is required in a more general situation. The same considerations hold for the work presented in [LKTT07], where Active Appearance Models are fit on the person face in order to discover which portion of a mall-shelf is observed.

In [LD08a], the visual field is modelled as a tetrahedron associated with a head pose detector. However, their model fixes the depth of the visual field, and this is quite unrealistic. SVF models the visual field as well, but in this case, owing to the 3D environment in which the SVF lives, one can let the SVF be bounded by the structure of the scene, which is more reasonable. Moreover, the proposed formulation is not restricted to controlled environments, but it can be employed to analyse any generic scene.

The proposal extends the work done in [FBMC09], which is the first to promote the use of the visual focus of attention for interaction modelling in a Computer Vision context.

6.6.2 Subjective View Frustum Estimation

The *Subjective View Frustum* (SVF) is defined as the polyhedron $\mathcal{D}$ depicted in Figure 6.33. It is composed of three planes that delimit the angles of view on the left, right and top sides, such that the angle span is 120° in both directions. The 3D coordinates of the points, corresponding to the head and feet of a subject are obtained from a multi-target tracker, while the SVF orientation is obtained from an head pose detector (see below). The proposed system is therefore composed of

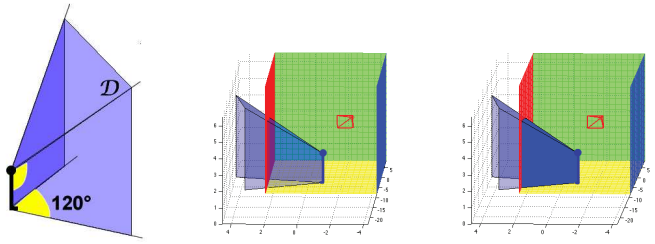


Fig. 6.33. Left: the SVF model. Centre: an example of SVF inside a 3D "box" scene. In red, the surveillance camera position: the SVF orientation is estimated with respect to the principal axis of the camera. Right: the same SVF delimited by the scene constraints (in solid blue).

four modules operating in cascade. First, the camera is calibrated and a (rough) 3D model of the scene is constructed. Second, a multi-target tracker detects people's position in each frame, and this data is used to guide the head pose detector. Finally, all the information is used to estimate the SVF. Each single module is detailed in the following.

3D Scene Estimation. Supposing that the camera monitoring the area is fully calibrated, the world reference system is put on the ground plane, with the z-axis pointing upwards. This allows to obtain the 3D coordinates of a point in the image if the elevation from the ground plane is known.

Therefore, a rough reconstruction of the area, composed of the principal planes present in the scene, can be carried out (an example in Fig. 6.33). This operation requires very little effort. In principle, a more detailed 3D map can be considered, if for example a CAD model of the scene is available or if a Structure-from-Motion algorithm [FFGT08] is applied. The choice depends on which level of detail one wants to gather from the SVF applications.

6.6.3 Tracking

Multi-target tracking has been well investigated in literature. In this work, a well-known method called Hybrid Joint-Separable (HJS) filter [Lan06] is used, because it deals with severe occlusions. It is essentially a multi-hypothesis particle filtering approach, able to sample the joint state space of the targets efficiently.

From the Bayesian perspective, the single object tracking problem aims at recursively calculate the posterior distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$, where $\mathbf{x}_t$ is the current state of the target (*e.g.*, its position), $\mathbf{z}_t$ is the current measurement or observation (e.g. the current frame), and $\mathbf{x}_{1:t}$ and $\mathbf{z}_{1:t}$ are the states and the measurements up to time t, respectively:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \propto p(\mathbf{z}_t|\mathbf{x}_t) \int p(\mathbf{x}_t|\mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{t-1}. \tag{6.67}$$

This recursive formulation is fully specified by the initial distribution $p(\mathbf{x}_0)$, the dynamical model $p(\mathbf{x}_t|\mathbf{x}_{t-1})$, and the observation model $p(\mathbf{z}_t|\mathbf{x}_t)$. Particle filtering approximates the posterior distribution by a set of N weighted particles, i.e.

$\{(\mathbf{x}_t^{(n)}, \mathbf{w}_t^{(n)})\}_{n=1}^N$; a large weight $\mathbf{w}_t^{(n)} \propto p(\mathbf{z}_t|\mathbf{x}_t^{(n)})$ mirrors a state $\mathbf{x}_t^{(n)}$ with high posterior probability. Hence, particle filtering consists in generating new hypothesis according to $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ and evaluating their likelihood $p(\mathbf{z}_t|\mathbf{x}_t)$.

HJS filter is an extension of this framework for multiple targets. It adopts the approximation $p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \prod_k p(\mathbf{x}_t^k|\mathbf{z}_{1:t})$, that is, the joint posterior

$$\mathbf{x}_t = \{\mathbf{x}_t^1, \mathbf{x}_t^2, \ldots, \mathbf{x}_t^K\}$$

could be approximated via the product of its marginal components (k indexes the individual targets). The dynamics and the observation models of HJS are marginalized out as follows:

$$p(\mathbf{x}_t^k|\mathbf{x}_{t-1}^k) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}^{\neg k}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{t-1:t}^{\neg k} \tag{6.68}$$

$$p(\mathbf{z}_t|\mathbf{x}_t^k) = \int p(\mathbf{z}_t|\mathbf{x}_t) p(\mathbf{x}_t^{\neg k}|\mathbf{z}_{1:t-1}) d\mathbf{x}_t^{\neg k} \tag{6.69}$$

where $\neg k$ means all the targets but the kth. These equations encode an intuitive strategy: the dynamics and the observation models of the kth target lie upon the consideration of a joint dynamical model $p(\mathbf{x}_t|\mathbf{x}_{t-1}) \approx p(\mathbf{x}_t) \prod_k q(\mathbf{x}_t^k|\mathbf{x}_{t-1}^k)$ and $p(\mathbf{z}_t|\mathbf{x}_t)$, respectively. The joint distribution $p(\mathbf{x}_t)$ avoids that multiple targets with single motion $q(\mathbf{x}_t^k|\mathbf{x}_{t-1}^k)$ collapse in a single location. $p(\mathbf{x}_t^k|\mathbf{x}_{t-1}^k)$ is different from $q(\mathbf{x}_t^k|\mathbf{x}_{t-1}^k)$, since $q(\mathbf{x}_t^k|\mathbf{x}_{t-1}^k)$ does not take into account the interactions between targets, unlike $p(\mathbf{x}_t^k|\mathbf{x}_{t-1}^k)$, which is integrated over $\mathbf{x}_{t-1:t}^{\neg k}$. The joint observation model considers that the visual appearance of a single target may be occluded by another object simulating a z-buffer. The two models are weighted by posterior distributions that essentially promote trusted joint objects configurations (not considering the kth object). For more details about how to compute Eq. 6.67, 6.68 and 6.69, the HJS algorithm and the features used for tracking refer to the original paper [Lan06].

6.6.4 Head Orientation Classification

The tracker provides the location of the head and the feet for each person in each frame. As for the head approximate position, I define a square window I of size $r \times r$, where I run the multi-class algorithm that recovers the head orientation. The size r has to be large enough to contain a head, considering the experimental physical environment and the camera position. ARCO 6.3 is adopted as head orientation classifier with the same settings adopted in Sec. 6.3.3.1 for what concerns the basic image features used to form ARCO, the learning framework, and its relative parameters.

In this case 5 classes named North, South, East, West, and Background are used. The first four classes indicate the four directions related to the camera orientation. The Background class manages the cases when the tracker fails in providing a correct head position. Actually, the use of only four directions may lead to rough estimates, but it should be considered that the resolution of the source video data is very poor.

The ARCO representation has several advantages. First, it allows to take into account different features, inheriting their expressivity and exploiting possible correlations, by using the covariance local descriptor; to such an extent that, it could be considered as a compact and powerful integration of features. Second, due to the use of integral images exploited in the computation of the covariance matrices [TPM08], ARCO is fast to compute, making it suitable for a possible real-time usage.

6.6.5 Subjective View Frustum

The SVF $\mathcal{D}$ is computed precisely using Computational Geometry techniques. It can be written as the intersection of three negative half-spaces defined by their supporting planes of the left, right and top sides of the subject, respectively. In principle, the SVF is not bounded in depth, modelling the human capability of focusing possibly on a remote point located at infinite distance. However, in practice, SVF is limited by the planes that set up the scene, according to the 3D scene (see Fig. 6.33). The scene volume is similarly modelled as intersection of negative half-spaces. Consequently, the exact SVF inside the scene can be computed solving a simple *vertex enumeration* problem, for which very efficient algorithms exist in literature [MS85].

6.6.6 The Inter-Relation Pattern Matrix

The SVF can be employed as a tool to discover the visual dynamics of the interactions among two or more people. This analysis relies on few assumptions with respect to social cues, i.e. that the entities involved in the *social interaction* stands closer than 2 meters (thus covering the *socio-consultive zone* – between 1 and 2 meters – the *casual-personal zone* – between 0.5 and 1.2 meters – and the *intimate zone* – around 0.4-0.5 meters) [VPB09]. Then, it is generally well-accepted that initiators of conversations often wait for visual cues of attention, in particular, the eye contact, before beginning a conversation during encounters [WFDJ94, LWB00, JWVG03]. In this sense, SVF may be employed in order to infer whether an eye contact occurs among close subjects or not. This happens with high probability when the following conditions are satisfied: 1) the subjects are closer than 2 meters; 2) their SVFs overlap, and 3) their heads are positioned inside the reciprocal SVFs (see Figure 6.34). In Fig. 6.34, a 2D projection of the 3D frustum is shown for illustrative purposes. Anyway, the real intersection is calculated between the genuine 3D SVFs. The Inter-Relation Pattern Matrix (IRPM) records when a possible social interaction occurs, and it can be formalized as a three-dimensional matrix [Fre89], where each entry $(i, j, t) = (j, i, t)$ is set to one, if i and j satisfy the three conditions above, during the t-th time instant.

The IRPM matrix is employed to analyse time intervals in which to look for social interactions. Suppose to focus on the time interval $[t-T+1, t]$ are considered. In this case, all the IRPM slices that fall in $[t-T+1, t]$, summing them along the t direction, and obtaining the *condensed* IRPM (cIRPM). Intuitively, the higher is the entry $\text{cIRPM}_t(i, j)$, the stronger is the probability that subjects i and j are related during the interval $[t-T+1, t]$. Therefore, in order to detect a relation

Fig. 6.34. Left: two people are talking to each other. Right: top view projection of their SVFs: the estimated orientation, East for 1 and West for 2, is relative to the camera orientation (the pyramid in red in the picture). The SVFs satisfy the three conditions explained in Sec. 6.6.6.

between a pair of individuals i, j in the interval $[t - T + 1, t]$, one can check if $\text{cIRPM}_t(i, j) > Th$, where Th is a threshold defined a priori. This threshold filters out noisy interaction detection: actually, due to the errors in the tracking and in the head pose estimation, the lower the threshold, the higher the possibility of false positive detections. In the experiments, how the choice of the parameters T and Th modifies the goodness of the results is shown, in term of social interaction detections.

The cIRPM represents one-to-one exchanges only, but the goal is also to capture if there are *groups* in the scene. The term group is used in its common definition, i.e. "an assemblage of objects standing near together, and forming a collective unity; a knot (of people), a cluster (of things)". The latter significance is closer to our aims.

Operationally, the $cIRP\mathbf{M}$ is treated as the adjacency matrix of a graph, with a vertex v_i for each people in the scene, and an edge e_{ij} if $cIRPM_t(i, j) > Th$. The *groups* present in the scene are detected by computing the connected components of the graph. Some illustrative examples are depicted in Fig. 6.38, 6.39, and 6.40.

6.6.7 Experimental Results

The experiments aim to show the capabilities of the proposed approach. First, the performance of tracking and head orientation classification are validated separately, in order to check the behaviour of the single modules. Then, it is shown how these modules perform grouped together, by analysing the employment of the IRPM, and its capability in individuating social exchanges.

Regarding the head orientation classification model, a multi-class classifier is built for head pose classification on the GDet head orientation classification dataset originally available in [Tosb]. This dataset is extracted from the GDet (Groups Detection) Dataset [Baz] that is composed of 12 sequences, each one lasting few minutes. The scenario is composed of a vending machines room where people take drinks and food, and chat. The videos have been obtained from two monocular cameras, located on a room corner close to the ceiling. The GDet head dataset is built using a training set extracted from a different video sequence. In fact, heads

Fig. 6.35. Examples of the GDet head orientation dataset. Each row corresponds to a different class.

are manually cropped from the GDet [Tosb] dataset and the QMUL head pose dataset [OGX09] are used to balance the number of examples in each class. All the images are divided into 5 classes 1555 Back, 1992 Front, 1990 Left, 2808 Right, and 2948 Background 20×20 pixels images. The testing set is composed of ROIs of variable sizes from the GDet scenario, which are manually classified as dome for the training set. In Fig. 6.35 some examples are reported.

The confusion matrix associated with the GDet head dataset are provided in Fig. 6.36 the confusion matrix for the enriched dataset.

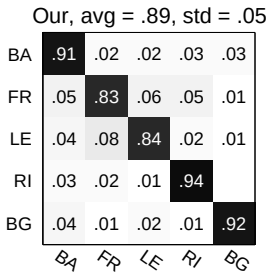


Fig. 6.36. The confusion matrix for the GDet head orientation classification dataset [Tosa].

Concerning the analysis of social exchanges, a video of about 3 hours and a half, portraying a vending machines area where students have coffee and discuss. The video footage was acquired with a monocular IP camera, located on an upper angle of the room. The people involved in the experiments were not aware of their aim, and behaved naturally. Afterwards, since creating the ground truth by using only the video is an hard task, were asked to some of them to fill a questionnaire inquiring if they had talked to someone in the room and to whom. Then, a video analysis was performed by a psychologist able to detect the presence of interactions between people. The questionnaires were used as supplementary material to confirm the validity of the generated ground truth. This offers a more trustworthy set of ground truth data for the experiments.

The original 3.5h video of the GDet [Baz] dataset has been reduced to this small set of sequences for several reasons: first, a lot of frames are empty, because the recording has been done early morning. Second, only the sequences where the

ground truth was evident and clear, i.e. all the components of each group were known are considered. Third, they were chosen such that to represent different situations, with people talking in groups[3] and other people not interacting with anyone.

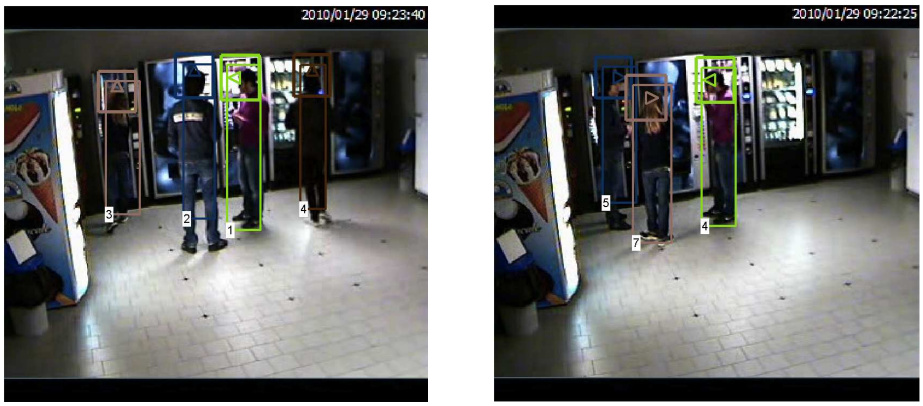


Fig. 6.37. Examples of tracking and head orientation classification results. The biggest box represents the tracking estimation, the smaller box the area where the head is positioned, and the triangle depicts the estimated head orientation.

For each subsequence, tracking is estimated, head orientation classification (some examples are shown in Figure 6.37) and the three-dimensional IRPM is built, that tells which people are potentially interacting at a specific moment. For the head classification part, the 4 Head Pose dataset is enriched with head images coming from the Vending Machine dataset, in order to enrich accuracy and robustness. About 150 images are added for each FG class, and 1840 images to the Background.

The results are compared to the ground truth. 8/12 sequences where correctly interpreted by the system. One can be considered wrong, because there are 2 groups in the scene, and our system reveals that they all belong to the same group. In the other three sequences there are some imprecisions, like a person left out of a group. These imprecisions are mainly due to error propagation from tracking and head orientation classification, particularly challenging when people are grouped and frequently intersect. A qualitative analysis of the results is shown in Figures 6.38, 6.39 and 6.40. The first row of each figure depicts three sampled frames from each sequence and contains the identifiers of each person. The second row depicts the cIRP**M**, on the left[4], and the graph structure that defines the group interactions on the right. In these three experiments, all the groups are detected correctly; Fig. 6.40 shows that the proposed model is able to detect interactions when the scene contains several groups.

A more sophisticated analysis of accuracy performances of the proposed method is shown in Fig. 6.41 and Fig. 6.42. The graphs summarize the group

[3] The groups are formed by 3 individuals, in average.
[4] Blue cells mean zeros. The values of the cIRP**M** below Th are discarded.

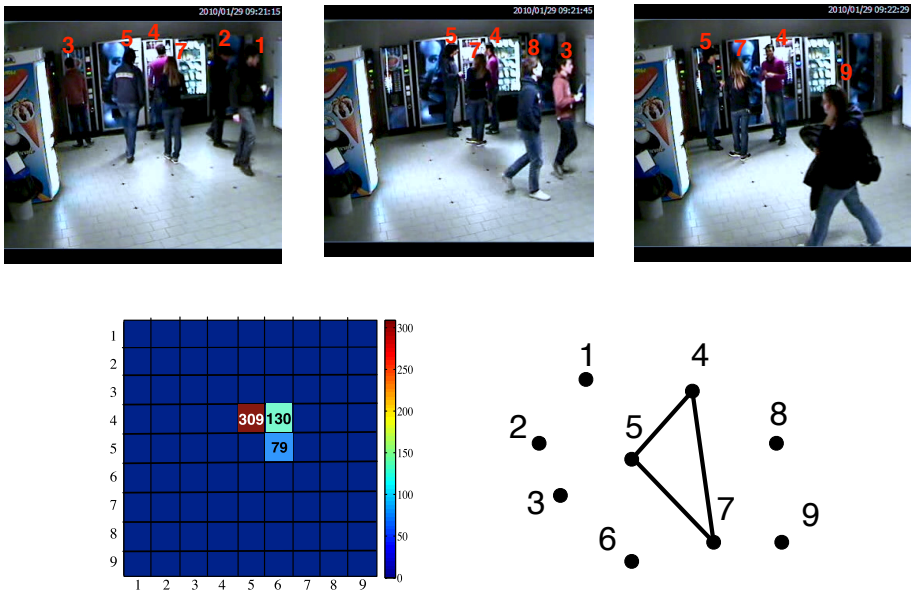


Fig. 6.38. Example of condensed IRPM analysis of sequence S_{04}. On the top row, some frames of the sequence. On the bottom row, on the left, the thresholded cIRPM matrix. Being the cIRPMs symmetric and having null main diagonals, for clarity only its strictly upper triangular part is reported. On the right, the correspondent graph. As one can notice, only one group (composed of people 4, 5 and 7) is detected. This is correct, since the other people of the sequence do not interact.

detection accuracy in terms of precision (on the left) and recall (on the right). In the definition of those measurements, true positive occur when a group is detected considering all its constitutive members. If a person that belongs to a group is not detected, a false negative appears, and a similar reasoning applies for the false positive.

Fig. 6.41 depicts the statistics obtained by increasing the size T of the time interval $[t-T+1, T]$ (x-axis) used to accumulate the IRPM. Each curve corresponds to a value of threshold Th (5, 20, 60 and 100). From Fig. 6.41 shows, first of all, increasing T gives worse accuracy. Moreover, the peak of each curve depends on both the threshold and the time interval size. The best performances by setting the Th equal to 20 obtained; the peak of this curve corresponds to a T equal to 300. Instead, Fig. 6.42 shows the performances increasing the threshold (x-axis) used to detect the groups. Each curve corresponds to a value of T (120, 300, 480, 720, 900, and 1200). The common behaviour of all the curves is that increasing and decreasing too much the threshold, the accuracy decreases. This analysis confirms that the best performances are given by setting the threshold to 20 and the time interval to 300. When T increases the accuracy drastically decreases and the peak of each curve is shifted, depending on the time interval size.

Intuitively, when the threshold is too low and the time window is too small, the proposed method detects interactions that could contain false positive. Increasing the size of the time window and the threshold permits to average these false

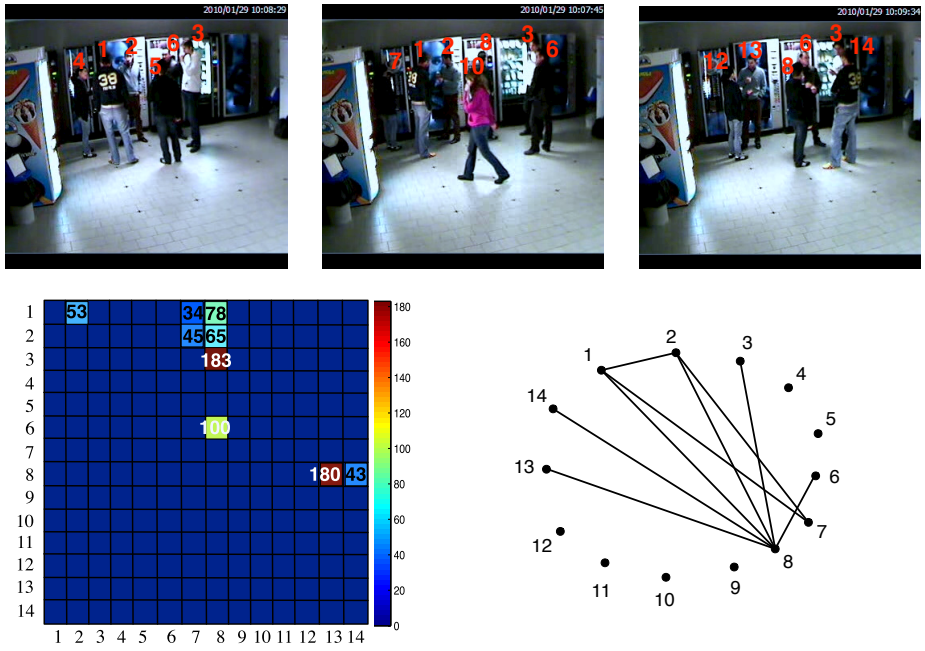


Fig. 6.39. Example of condensed IRPM analysis of sequence S_{08}. On the top row, some frames of the sequence. On the bottom row, on the left, the threshold cIRPM matrix. On the right, the correspondent graph. One big group (1,2,3,6,7,8,13,14) is detected. Some people are represented by more than one track, because of severe or complete occlusions the tracks are sometimes lost and reinitialized. The group selected is correctly composed of the people associated to the labels. Another person (10) enters in the room and does not interact. The same behaviour is witnessed in the cIRPM.

positives out and cancel them out, because the IRPM becomes more stable. On the other hand, when the threshold is too high, the proposed model is not able to detect interactions, because $cIRPM_T(i,j) > Th$ is zero for each (i,j). To deal with this problem, the time interval can be made larger. However, in this case, a group interaction interval could be smaller than the time window, and in any case the threshold is too high to detect groups. For these reasons, precision and recall in Fig. 6.41 and Fig. 6.42 decrease before and after the optimal setting of the parameters ($Th = 20$ and $T = 300$).

6.7 Object Classification using Tensors

This Section aims to understand if it is possible to exploit the tensor representation for classification purposes and to build a more powerful object descriptor than COV (Covariance), exploiting more complex object representation than Chap. 4. In this Section, a learning framework is introduced to deal with different tensors, namely EMI (Entropy-Mutual Information) tensor and COV tensor.

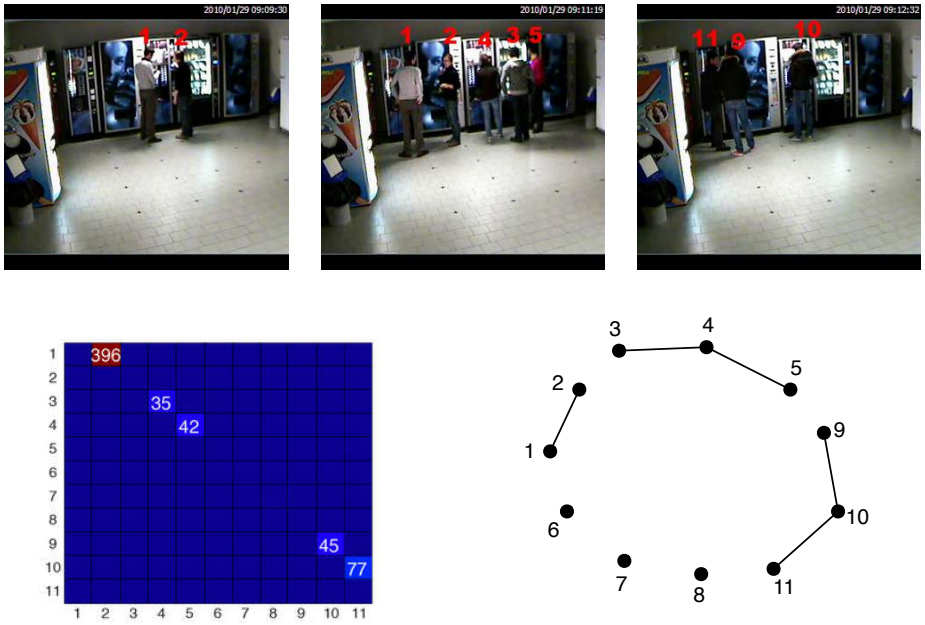


Fig. 6.40. Example of condensed IRPM analysis of sequence S_{01}. On the top row, some frames of the sequence. On the bottom row, on the left, the thresholded cIRPM matrix. On the right, the correspondent graph. Three groups (1,2),(3,4,5), and (9,10,11) are detected. Some people are represented by more than one track, because of severe or complete occlusions the tracks are sometimes lost and reinitialized (e.g. 6,7,8 are reinitialized as 9,10,11, respectively).

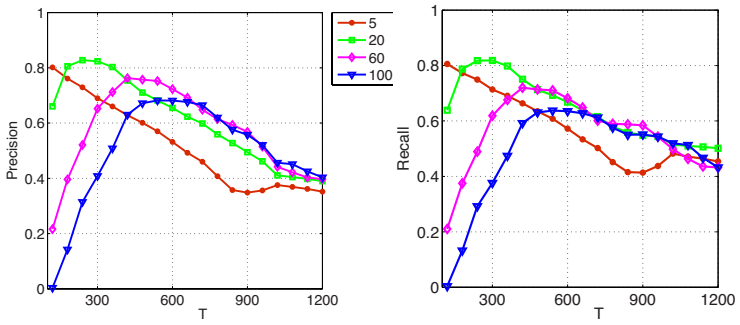


Fig. 6.41. Evaluation of precision (left) and recall (right) of the proposed method varying the size of the time interval $[t-T+1, t]$ (x-axis) used to compute the IRPM. The graph shows one curve for each threshold (5, 20, 60 and 100). The maximum both for the statistics is given by setting $Th = 20$.

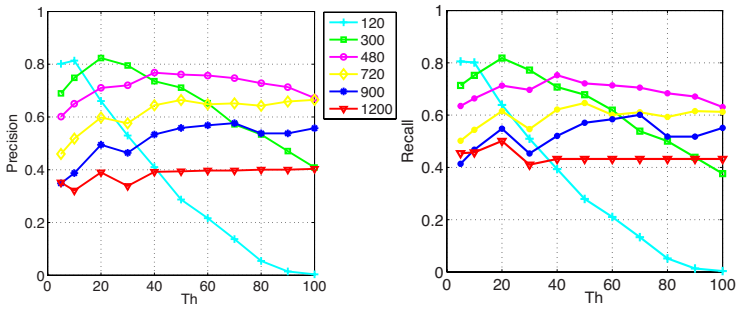


Fig. 6.42. Evaluation of precision (left) and recall (right) of the proposed method varying the threshold Th (x-axis) used to detect the groups. The graph shows one curve for each time window (120, 300, 480, 720, 900, and 1200). The maximum both for the statistics is given by setting $T = 300$ and the peak is where $Th = 20$ (also according to Fig. 6.41).

The Section is organized as follows: in Sec. 6.7.1 the object model adopted for the experiments is described in detail and in Sec. 6.7.2 a comparative evaluation of EMI and COV on different classification tasks is shown.

6.7.1 Object Models for General Classification Problems

I recall that in Sec. 4.3 an image was represented by one tensor instance to study the performances of the different tensors without using complex object model. However, in order to maximize the classification accuracy, a more complex object representation has to be adopted. To build a sufficiently general, yet discriminative, descriptor, the idea proposed in [BZM07a] is utilized. Therefore, a pyramidal patch-based representation is used. In particular, each image is divided into a sequence of increasingly finer spatial grids by doubling the number of divisions in each axis direction repeatedly. The cell counts at each level of resolution are the bin counts for the histogram representing that level. A 3 level pyramid is adopted.

In order to make a fair comparison, the same structure is adopted for all the matrix tensors (i.e. EMI and COV). The basic image feature sets used to build the matrix tensors will be detailed in the experimental Section, but they are quite similar to the ones used in Sec. 4.3.

6.7.2 A Comparative Experimental Study

In this Section, a comparative study on different public available datasets for the object classification task is described. As done in Sec. 4.3, a kernel SVM is used as learning framework (see Alg. 7 for details). For what concerns the learning parameter C, the grid-search is applied varying C in $2^{-3}, \ldots, 2$ with step 1.

LabelMe. The annotated LabelMe [RTMF08] dataset is exploited to test the ability of the tensor representations to discriminate among fine categories, such as legs and arms. LabelMe is a database and an on-line annotation tool that allows to share images and annotations. It is designed for object class recognition and contains various object classes. From this dataset, only 4 different object classes

are extracted, all belonging to the same object as one can see in Fig. 6.43. The classes are 4 human body parts: arm, head, leg and torso. Images are reflected building a dataset of 16288 examples. Also in this case, as for Pascal VOC 2009,

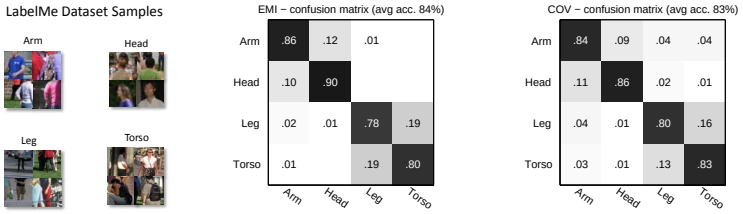


Fig. 6.43. Some examples and Confusion Matrices (CMs) for the LabelMe [RTMF08] dataset. On the left, the CM given using the EMI tensor, while on the right the CM associated with the COV tensor.

a 5-fold cross-validation procedure has been used. During each training phase, 2000 randomly selected examples per class populate the training set and all the remaining are used for testing purposes. Each example is described with the feature set of Eq. (6.65) and, again, one tensor is used to describe the image of an object. In Fig. 6.43 the CMs of EMI and COV tensors are shown. It is clear that EMI outperforms COV also in this finer classification task. Moreover, since the classes are highly overlapped, EMI better manages the presence of noise in images. This is probably due to the fact that it uses the histogram intermediate representation that improves the description robustness, if compared to COV tensors.

Pascal VOC 2009. This dataset [EVGW$^+$] consists of a few 17895 high resolution images annotated with bounding boxes for objects of twenty categories (e.g. car, bus, aeroplane, ...). The goal of this challenge is to classify objects in realistic scenes (i.e. not pre-segmented objects). Basically, it is a supervised learning problem where a training set of labelled images is provided. In this case the results of the SST tensors cannot be provided because the images of the dataset have a variable size.

In Fig. 6.44 the best confusion matrices for EMI and COV are reported. In

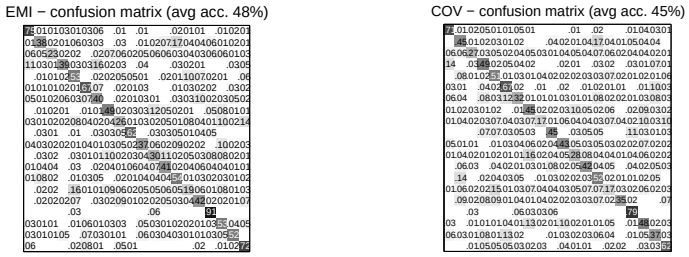


Fig. 6.44. Confusion Matrices (CMs) for the Pascal VOC 2009 [EVGW$^+$] dataset. On the left the CM given using the EMI tensor, while on the right the CM associated with the COV tensor.

this experiment, EMI clearly outperforms the COV representation with an average accuracy of 48%, against a 45% provided by the COV tensor.

Also testing tensor representations, in function of the images resolution, can be interesting. Using the bilinear re-sampling function, provided by Dollar toolbox [Dol], all the Pascal's images are down-sampled. As one can see in Fig. 6.45, two different kinds of down-sampling are adopted: in the first case it operates without preserving the image size, while in the second case it does. This is due to the fact that it is interesting to study the behaviour of the tensor in function both of image size and of resolution.

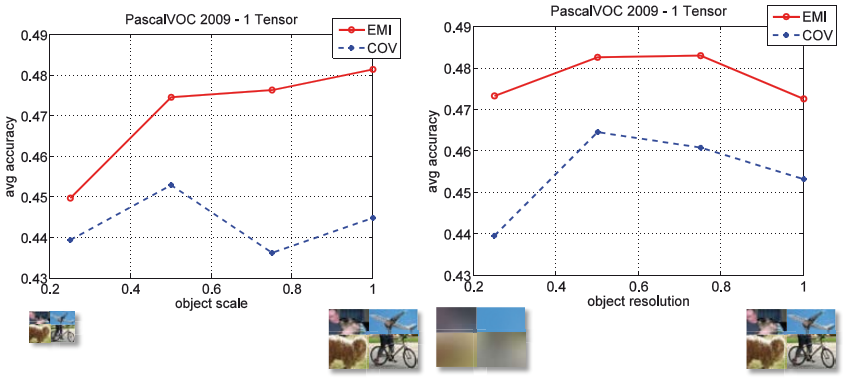


Fig. 6.45. Classification performances of EMI and COV tensors on Pascal VOC 2009 in terms of mean classification accuracy varying objects scale and resolution.

CIFAR10. The CIFAR10 dataset [KH06] is a hand-labelled subset of a larger dataset of 80 million tiny images. These images were downloaded from the Internet and down-sampled to 32×32 pixels. The CIFAR10 subset has 10 object

Fig. 6.46. Example of images in the CIFAR10 dataset.

categories, namely aeroplane, bird, car, cat, deer, dog, frog, horse, ship, and truck (see Fig. 6.46). The training set has 5000 examples per class, the test set has 1000 examples per class. The low resolution and variability make recognition very diffi-

cult and a traditional method based on features extracted at interest points does not work.

Since the recognition task on this dataset is hard, the feature description is enhanced using the pyramidal descriptor presented in Sec. 6.7.1 that adds 2 sub-layers to the single (top layer) descriptor utilized before. For each patch of that pyramidal structure a tensor is extracted, vectorized and concatenated. The dimension of the final object descriptor is clearly larger if compared to the one obtained using only one tensor for object description. Therefore PCA (Principal Component Analysis) is applied to reduce automatically the dimensionality of the final object description [ZLY10]. The optimal feature descriptor dimensionality is established fixing to 96% the data energy that should be preserved after the linear projection. That procedure is used both for EMI and for COV tensors. Tab. 6.7 reports a comparison using EMI and COV tensors on CIFAR images resized at a resolution of 128×128. Different feature sets, already implemented in the Dollar's toolbox [Dol], have been applied. The first filter bank has been already presented in Eq. (6.65). It is composed of a set of 8 DOOG filters and other Gradient and color features. This feature set is named DOOG in Tab. 6.7. Replacing the filter set with a different filter bank from Serge Belongie [MBLS01] composed of 40 filters, a much more informative filter representation called Belongie is reported in Tab. 6.7. Is is possible to observe that the pyramidal EMI representation combined

Tensor Representation	Filters' Set	Avg Accuracy
EMI	Belongie	**52**%
EMI	DOOG	49%
COV	Belongie	40%
COV	DOOG	38%

Table 6.7. Test recognition accuracy on the CIFAR10 dataset produced by different pyramidal tensor representations.

with Belongie filter set offers the best performances, outperforming the COV representation. To consolidate that result, the comparison between EMI and COV is made on a much more difficult dataset in the next experiment.

CIFAR100. CIFAR100 dataset [KH06], as CIFAR10, is a hand-labelled subset of a larger dataset of 80 million tiny images. Also in this case images were downloaded from the Internet and down-sampled to 32×32 pixels. CIFAR100 is composed of 100 categories of objects. Its training set and its testing set have both 100 examples per class. The same experimental setting as CIFAR10 is adopted, as described above. In Tab. 6.8 the experimental results are reported. As for CIFAR10 the best average accuracy is obtained using pyramidal EMI tensor and Belongie filter set, which confirms the superiority of EMI on COV tensor representation.

Tensor Representation	Filters' Set	Avg Accuracy
EMI	Belongie	**32**%
EMI	DOOG	26%
COV	Belongie	19%
COV	DOOG	18%

Table 6.8. Test recognition accuracy on the CIFAR100 dataset produced by different pyramidal tensor representation.

7

Conclusions

This thesis focuses on the challenging task of classification and detection of defined classes of objects. For those tasks an important aspect has to be considered to obtain good performances in terms of classification and detection accuracy, that is how to represent a visual object. To this end, novel tensor representations are proposed and investigated, able to combine multiple sources of information simultaneously. Then the models to describe objects in problematic scenarios like in the video surveillance context are designed and implemented. Moreover, different tensor learning frameworks are presented, based on the problem setting: object detection, classification, and regression.

More technically speaking, a study of how to represent objects using tensors is outlined. It is inspired by the successful performances achieved by covariance tensors [FPAA07, TPM08], that are used to represent visual objects. Therefore, four different tensor representations that I called Entropy-Mutual Information (EMI), Self-Similarity Structure Tensor ($\text{SST}_{\text{struct}}$), Self-Similarity Content Tensor ($\text{SST}_{\text{content}}$), and Grassmann Tensor (GRT) are proposed. Depending on the task considered (detection or classification), those lead to better performances, if compared with covariance (COV) tensors.

EMI tensor is composed of mixing entropy and mutual information and shows its potentiality in object classification problems where it outperforms COV representation. $\text{SST}_{\text{content}}$, which measures the distance among different features, is more lighter and efficient respect to COV and EMI. It permits to combine many image features together with a very low computational cost. Therefore, using a large set of image features, it shows to lead to better performances than EMI and COV for general classification problems. $\text{SST}_{\text{struct}}$ measures the self-similarity of an object composed of parts; it uses the structural information to discriminate an object. $\text{SST}_{\text{struct}}$ leads to state-of-the-art performances on the DaimlerChrysler dataset [MG06] for the low resolution pedestrian detection. Regarding GRT, similarly to the structural SST, is used to characterise the structure of an object as set of vectors instead of the matrix representation of $\text{SST}_{\text{struct}}$. GRT outperform $\text{SST}_{\text{struct}}$ in terms of classification accuracy, but its computational cost is high.

Future research on tensor representations will check whether other tensor representations, which are well known inside and outside vision and have been studied in fields such as physics and robotics, can be used to describe visual objects. These

representations and the associated (manifold) geometry can be analysed for practical applications in computer vision problems based on the learning frameworks proposed in this thesis. Moreover, it can be interesting to study the combination of different tensor representations: for example, combining EMI and GRT could give the best performances of all the classification datasets used in this thesis. Unfortunately, the computational cost can be high, so some techniques to compute tensors and their combination efficiently must be studied.

For what concerns the problem of object detection, the thesis focuses on the pedestrian detection drawing particular attention to build robust detectors and exploiting tensors that can be efficiently computed (i.e. COV and $\mathrm{SST}_{\mathrm{struct}}$). For example, COV is used to improve the state-of-the-art method [TPM08] for pedestrian detection, allowing also an estimation of occlusions in a fine way. COV can be also exploited to represent a pedestrian by its single body parts building part-based human detector. In this case the learning framework combines the weighted boosted responses of part detectors, and indicates the upper part of the body as the best part usable to capture human beings. The resulting framework is light and robust and it sets the state-of-the-art on the INRIA Person dataset [Dal05]. Another light detection architecture proposed in this thesis, exploits the hardware acceleration to boost the efficiency in the usage of COV tensors extracted from a regular grid of image patches. This without loosing detection capability respect to the previous pedestrian detectors. In this way, this light framework can be implemented into an embedded device, in particular on an FPGA board. Every single part of the proposed framework (structure, classification approach, and features) is easily applicable also to different detection and classification problems like the ones in [FHT00, Bre84, TPM06].

Some of the above-mentioned pedestrian detection frameworks are implemented and applied into the SAMURAI system [sam], which contains robust moving object, segmentation, categorisation and tagging in video captured by multiple cameras from medium-long range distance.

For what regards the classification and regression problems, the thesis concentrates on the delineation of some descriptors based on the COV and EMI tensors and the relative learning frameworks. The general-purpose ARCO (ARray of COvariance matrices, adopts a theoretical framework of multi-class classification on Riemannian manifold Sym_d^+, leading to two remarkable advancements. From a practical point of view, ARCO can describe faces as well as pedestrians, by including arbitrary image features, and exploiting their dependencies via spatially local COV tensors. From a theoretical point of view, it is shown the fact that Sym_d^+ has non-positive sectional curvature and that the curvature is almost flat in some of its areas. Therefore, one can perform multi-class discriminative learning projecting the ARCO features on a tangent space at any point of Sym_d^+. The experimental Section validates the proposed approach, with novel state-of-the-art performances. Moreover, ARCO is applied as a part of a framework to compute the Subjective View Frustum (SVF), which may help understand social signals in a scene. It encodes the visual field of a person in a 3D environment. The SVF permits to define novel analysis tools, such as the Inter-Relation Pattern Matrix. Convincing results are shown, that lead to several future improvements: together with a refinement of the head pose detector (in order to find tilt and roll parameters and a more

informative pan quantization), it may be also possible to jointly investigate gesture recognition modules, useful to capture different and more complicated social interactions. To this end, ARCO is improved both from the theoretical and the applicative point of view, turning it into WARCO (Weighted ARCO). WARCO is used to characterize tiny images of pedestrians in a surveillance scenario, specifically, to perform head orientation and body orientation estimation. The achieved results indicate that the framework will be adopted as standard tool in surveillance applications. Moreover, WARCO is valuable beyond the aim of the contingent application. In fact, a theoretically sound way to deal with covariance matrices is suggested, i.e. like they were points lying on a Euclidean space. This is possible thanks to a measure to approximate geodesic distances, the CBH1 measure, that works better than the standard Euclidean distance. Future research on this topic will check whether the triangular inequality holds for CBH1, in order to validate CBH1 as genuine distance. Furthermore, WARCO will be extended to became action descriptor, including the temporal dimension in the analysis. Moreover the theoretical analysis which has produced the CBH1 measure can be easily instantiated for all the symmetric spaces, like Grassmann manifolds. Nevertheless, since WARCO is combined with a kernel SVM framework, it has some computational limitations on large datasets. Therefore, to maximize its efficiency, FWARCO (Fast WARCO), for fast and robust inference, is introduced. It is based on Random Forest which has proven its efficiency and robustness on several classification and regression tasks. In fact, RF can be trained on large datasets with a low computational cost and without being affected by significant overfitting. In the same vein of [FGMR10], FWARCO has been combined to a hard negative mining strategy designed for RF. The result is an enhancement of the efficiency and the robustness of WARCO.

In this thesis two computer vision problems are faced, the detection and classification of defined classes of objects, also in connection with the object representation issue. Different models are presented and some novel descriptors are outlined. Notwithstanding, on one hand future research on tensor representations will reveal if other tensor representations, well known inside and outside vision and already studied in physics and robotics, can be utilized for visual object description. On the other hand, future research on tensors will prove if the theoretical framework proposed in this thesis to measure the distance among tensors living in symmetric spaces is a good choice. All of those are challenging issues to deal with. This thesis, far from exhausting the topics which tackles, paves the way for further research.

A

Publications

Book Chapter

- L. Bazzani, M. Cristani, G. Pagetti, **D. Tosato**, G. Menegaz, and V. Murino. Analyzing groups: a social signaling perspective. In Video Analytics for Business Intelligence, Studies in Computational Intelligence. Springer-Verlag, 2012. in press.

Journal

- **D. Tosato**, M. Spera, M. Cristani, and V. Murino, Characterizing humans on Riemannian manifolds, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), submitted 2011.
- L. Bazzani, **D. Tosato**, M. Cristani, M. Farenzena, G. Pagetti, G. Menegaz, and V. Murino. Social interactions by visual focus of attention in a three-dimensional environment. Expert Systems, 2012. in press.

Conference

- M. Cristani, L. Bazzani, G. Pagetti, A. Fossati, **D. Tosato**, A. Del Bue, G. Menegaz, and V. Murino. Social interaction discovery by statistical analysis of f-formations. In British Machine Vision Conference (BMVC), 2011. *Oral presentation.*
- S. Martelli, **D. Tosato**, M. Cristani, and V. Murino, Fast FPGA-Based Architecture for Pedestrian Detection Based on Covariance Matrices, IEEE International Conference on Image Processing (ICIP), 2011.
- S. Martelli, **D. Tosato**, M. Cristani, and V. Murino, FPGA-Based Pedestrian Detection Using Array of Covariance Features, ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC) 2011 (Oral Presentation). *Oral presentation.*
- **D. Tosato**, M. Farenzena, M. Cristani, M. Spera, V. Murino Multi-class Classification on Rie- mannian Manifolds for Video Surveillance K. Daniilidis, P. Maragos, N. Paragios (Eds.): ECCV 2010, Part II, LNCS 6312, pp. 378-391, 2010. Springer-Verlag Berlin Heidelberg 2010. *Oral presentation.*

- **D. Tosato**, M. Farenzena, M. Cristani, V. Murino, Part-based human detection on Riemannian Manifolds , IEEE International Conference on Image Processing (ICIP), 2010. *Oral presentation.*
- **D. Tosato**, M. Farenzena, M. Cristani, V. Murino, A Re-evaluation of Pedestrian Detection on Riemannian Manifolds , IRAP International Conference on Pattern Recognition, 2010.
- S. Martelli, **D. Tosato**, M. Farenzena, M. Cristani, and V. Murino, An FPGA-based Classification Architecture on Riemannian Manifolds, International Conference on Database and Expert Systems Applications Workshop: Interactive Multimodal Pattern Recognition in Embedded Systems (IMPRESS), 2010. *Oral presentation. Best paper award.*
- M. Farenzena, A. Tavano, L. Bazzani, **D. Tosato**, G. Pagetti, G. Menegaz, V. Murino, and M. Cristani. Social interaction by visual focus of attention in a three-dimensional environment. In Workshop on Pattern Recognition and Artificial Intelligence for Human Behavior Analysis at AI*IA, 2009.
- U. Castellani, L. Bazzani, **D. Tosato**, V. Murino, G. Rambaldelli, C. Perlini, M. Atzori, M. Tansella, and P. Brambilla. A learning by example approach for MRI analysis of human brain in the context of mental health. In ISMRM Annual Meeting, Berlin, Germany, 2007.

Technical Report

- **D. Tosato**, M. Cristani, V. Murino, S. Gong, and T. Xiang, Tensor Representations for Object Classification and Detection, Queen Mary University of London Press, September 2011.

References

[AA04] A. Ahmedsaid and A. Amira. Accelerating SVD on reconfigurable hardware for image denoising. In *Proc. ICIP*, volume 1, pages 259–262. IEEE, 2004.

[ACW+07] N. Archip, O. Clatz, S. Whalen, D. Kacher, A. Fedorov, A. Kot, N. Chrisochoides, F. Jolesz, A. Golby, P.M. Black, et al. Non-rigid alignment of pre-operative MRI, fMRI, and DT-MRI with intra-operative MRI for enhanced visualization and navigation in image-guided neurosurgery. *Neuroimage*, 35(2):609–624, 2007.

[AFPA05] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. Fast and simple calculus on tensors in the log-euclidean framework. In *Proc. MICCAI*, pages 115–122. Springer, 2005.

[AFPA08] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. Geometric means in a novel vector space structure on symmetric positive-definite matrices. *JMAA*, 29(1):328, 2008.

[AMS08] P.A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton Univ Pr, 2008.

[AR92] N. Ambady and R. Rosenthal. Thin slices of expressive behavior as predictors of interpersonal consequences: A meta analysis. *Psychological bulletin*, 111(2):256–274, 1992.

[ARS09] M. Andriluka, S. Roth, and B. Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *Proc. CVPR*, volume 1, pages 1014–1021, 2009.

[AT06] A. Agarwal and B. Triggs. A local basis representation for estimating human pose from cluttered images. In *Proc. ACCV*, pages 50–59. Springer, 2006.

[Baz] L. Bazzani. GDet (Groups Detection) Dataset. `http://www.lorisbazzani.info/code-datasets/`.

[BBBK08] A.M. Bronstein, M.M. Bronstein, M. Bronstein, and R. Kimmel. *Numerical geometry of non-rigid shapes*. Springer-Verlag New York Inc, 2008.

[BdFL+11] L. Bazzani, N. de Freitas, H. Larochelle, V. Murino, and J-A Ting. Learning attentional policies for object tracking and recognition in video with deep networks. In *Proc. ICML*, 2011.

[BDTB08] B. Babenko, P. Dollár, Z. Tu, and S. Belongie. Simultaneous Learning and Alignment: Multi-Instance and Multi-Pose Learning. In *Proc. Faces in Real-Life Images*, October 2008.

[Bel06] C.M. Bishop and SpringerLink (Service en ligne). *Pattern recognition and machine learning*, volume 4. springer New York, 2006.

[Ber03] M. Berger. *A panoramic view of Riemannian geometry*. Springer Verlag, 2003.

[BHHW05] A. Bar-Hillel, T. Hertz, and D. Weinshall. Object Class Recognition by Boosting a Part-Based Model. In *Proc. CVPR*, pages 702–709, 2005.

[BHLKG10] A. Bar-Hillel, D. Levi, E. Krupka, and C. Goldberg. Part-based feature synthesis for human detection. *Proc. ECCV*, I:127–142, 2010.

[BHW11] M. Brown, G. Hua, and S. Winder. Discriminative learning of local image descriptors. *IEEE Trans. PAMI*, 33(1):43–57, 2011.

[Bis05] C.M. Bishop. *Neural networks for pattern recognition*. Oxford Univ Pr, 2005.

[BJE+08] Y. Benezeth, P.M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger. Review and evaluation of commonly-implemented background subtraction algorithms. In *Proc. ICPR*, pages 1–4, 2008.

[BM09] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *Proc. CVPR*, pages 1365–1372. IEEE, 2009.

[BMF04] M. Bicego, V. Murino, and M.A.T. Figueiredo. Similarity-based classification of sequences using hidden Markov models. *PR*, 37(12):2281–2291, 2004.

[BML+08] I. Bravo, M. Mazo, J.L. Lázaro, P. Jiménez, A. Gardel, and M. Marrón. Novel HW architecture based on FPGAs oriented to solve the eigen problem. *IEEE Trans. VLSI*, 16(12):1722–1725, 2008.

[BO05] S.O. Ba and J.M. Odobez. Evaluation of multiple cue head pose estimation algorithms in natural environements. In *Proc. ICME*, pages 1330–1333. IEEE, 2005.

[BO06] S.O. Ba and J.M. Odobez. A study on visual focus of attention recognition from head pose in a meeting room. In *Proc. MLMI*, pages 75–87, 2006.

[BP09] S.R. Bulò and M. Pelillo. A game-theoretic approach to hypergraph clustering. *Advances in Neural Information Processing Systems*, 22:1–9, 2009.

[BR09] B. Benfold and I. Reid. Guiding Visual Surveillance by Tracking Human Attention. In *Proc. BMVC*, September 2009.

[Bre84] L. Breiman. *Classification and regression trees*. Chapman & Hall/CRC, 1984.

[Bre96] L. Breiman. Bagging predictors. *ML*, 24(2):123–140, 1996.

[Bre97] G.E. Bredon. *Topology and geometry*, volume 139. Springer, 1997.

[Bre01] L. Breiman. Random forests. *ML*, 45(1):5–32, 2001.

[BWBM06] T. Brox, J. Weickert, B. Burgeth, and P. Mrázek. Nonlinear structure tensors. *JIVC*, 24(1):41–55, 2006.

[BZM07a] A. Bosch, A. Zisserman, and X. Munoz. Representing shape with a spatial pyramid kernel. In *Proc. ICIVR*, pages 401–408. ACM, 2007.

[BZM07b] A. Bosch, A. Zisserman, and X. Muoz. Image classification using random forests and ferns. In *Proc. ICCV*, pages 1–8. IEEE, 2007.

[CBP+11] M. Cristani, L. Bazzani, G. Paggetti, A. Fossati, A. Del Bue, D. Tosato, G. Menegaz, and V. Murino. Social interaction discovery by statistical analysis of F-formations. In *Proc. BMVC*, 2011.

[CCFM08] U. Castellani, M. Cristani, S. Fantoni, and V. Murino. Sparse points matching by combining 3D mesh saliency with statistical descriptors. In *Proc. CGF*, volume 27, pages 643–652. Wiley Online Library, 2008.

[Cha06] I. Chavel. *Riemannian Geometry - A modern introduction*. Cambridge University Press, Cambridge, 2006.

[CL] C.C. Chang and C.J. Lin. LIBSVM: a library for support vector machines. `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`.

[CNM06] R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proc. ICML*, pages 161–168. ACM, 2006.

[CP02] T. Choudhury and A. Pentland. The sociometer: A wearable device for understanding human networks. In *Proc. CSCW - Workshop on ACCUCE*, 2002.

[CPV+11] M. Cristani, A. Pesarin, A. Vinciarelli, M. Crocco, and V. Murino. Look at Who's Talking: Voice Activity Detection by Automated Gesture Analysis. In *Proc. InterHub*, 2011.

[CRCZ05] R. Chellappa, A.K. Roy-Chowdhury, and S.K. Zhou. *Recognition of Humans and Their Activities Using Video*. Morgan & Claypool Publishers, 2005.

[CSK11] A Criminisi, J Shotton, and E Konukoglu. Decision Forests for Classification, Regression, Density Estimation , Manifold Learning and Semi-Supervised Learning. Technical report, Microsoft Research, 2011.

[CST04] N. Cristianini and J. Shawe-Taylor. *Kernel methods for pattern analysis*. Cambridge University Press, 2004.

[CV09] H.E. Cetingul and R. Vidal. Intrinsic mean shift for clustering on Stiefel and Grassmann manifolds. In *Proc. CVPR*, pages 1896–1902. IEEE, 2009.

[CVV10] M. Cristani, V.Murino, and A. Vinciarelli. Socially intelligent surveillance and monitoring: Analysing social dimensions of physical space. In *Proc. SISM Workshop*, San Francisco, California, 2010.

[Dal05] N. Dalal. `http://pascal.inrialpes.fr/data/human/`, 2005.

[DB08] M. Donoser and H. Bischof. Using covariance matrices for unsupervised texture segmentation. In *Proc. ICPR*, pages 1–4, 2008.

[DBB+08] P. Dollár, B. Babenko, S. Belongie, P. Perona, and Z. Tu. Multiple component learning for object detection. In *Proc. ECCV*, pages 211–224. Springer, 2008.

[DC92] M.P. Do Carmo. *Riemannian geometry*. Birkhauser, 1992.

[DJ94] M.P. Dubuisson and A.K. Jain. A modified hausdorff distance for object matching. In *Proc. ICPR*, volume 1, pages 566–568. IEEE, 1994.

[DK00] J.J. Duistermaat and J.A.C. Kolk. *Lie groups*. Springer Verlag, 2000.

[Dol] Piotr Dollár. Piotr's Image and Video Matlab Toolbox (PMT). `http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html`.

[DT05] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. CVPR*, volume 1, page 886, 2005.

[DTPB09] P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral channel features. In *Proc. BMVC*, 2009.

[DTTB07] P. Dollár, Z. Tu, H. Tao, and S. Belongie. Feature Mining for Image Classification. In *Proc. CVPR*, pages 1–8, 2007.

[DWSP09] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *Proc. CVPR*, pages 304–311. IEEE, 2009.

[DWSP11] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian Detection: An Evaluation of the State of the Art. *IEEE Trans. PAMI*, 1(99):1–20, 2011.

[EG09] M. Enzweiler and M. D. Gavrila. Monocular Pedestrian Detection: Survey and Experiments. *IEEE Trans. PAMI*, 31:2179–2195, 2009.

[EG10] M. Enzweiler and D.M. Gavrila. Integrated pedestrian classification and orientation estimation. In *Proc. CVPR*, pages 982–989. IEEE, 2010.

[EVGW+] M. Everingham, L. Van Gool, CKI Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2009 Results. http://www.pascal-network.org/challenges/VOC/voc2009.

[FBMC09] M. Farenzena, L. Bazzani, V. Murino, and M. Cristani. Towards a subject-centered analysis for automated video surveillance. In *Proc. ICIAP*, 2009.

[FCH+08] R.E. Fan, K.W. Chang, C.J. Hsieh, X.R. Wang, and C.J. Lin. LIBLINEAR: A library for large linear classification. *JMLR*, 9:1871–1874, 2008.

[Fel01] P.F. Felzenszwalb. Learning models for object recognition. In *Proc. CVPR*, volume 1, pages I–1056. IEEE, 2001.

[FFGT08] M. Farenzena, A. Fusiello, R. Gherardi, and R. Toldo. Towards unsupervised reconstruction of architectural models. In *Proc. VMV*, pages 41–50, 2008.

[FGMR10] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Trans. PAMI*, 32(9):1627–1645, 2010.

[FGVG11] G. Fanelli, J. Gall, and L. Van Gool. Real time head pose estimation with random regression forests. In *Proc. CVPR*, pages 617–624. IEEE, 2011.

[FHT00] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *The annals of statistics*, 28(2):337–374, 2000.

[Fis] R. Fisher. CAVIAR Case Scenarios. `http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/`.

[FK97] A.T. Fomenko and T. Kunii. *Topological modeling for visualization*. Springer Verlag, 1997.

[FLPJ04] P.T. Fletcher, C. Lu, S.M. Pizer, and S. Joshi. Principal geodesic analysis for the study of nonlinear statistics of shape. In *Proc. ECCV Workshop*, volume 23, pages 995–1005. IEEE, 2004.

[FNHB05] A.R.J. Francois, R. Nevatia, J. Hobbs, and R.C. Bolles. VERL: An Ontology Framework for Representing and Annotating Video Events. *IEEE Trans. MM*, 12:76–86, 2005.

[FPAA07] P. Fillard, X. Pennec, V. Arsigny, and N. Ayache. Clinical DT-MRI estimation, smoothing, and fiber tracking with log-Euclidean metrics. *IEEE Trans. MI*, 26(11):1472–1482, 2007.

[Fre89] L. Freeman. Social networks and the structure experiment. In *Proc. Research Methods in Social Network Analysis*, pages 11–40, 1989.

[Fre95] Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995.

[Fre01] Y. Freund. An adaptive version of the boost by majority algorithm. *ML*, 43(3):293–318, 2001.

[FS97] Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *JCSS*, 55(1):119–139, 1997.

[FV06] L.M. Fuentes and S.A. Velastin. People tracking in surveillance applications. *JIVC*, 24(11):1165 – 1171, 2006.

[FVJ08] P.T. Fletcher, S. Venkatasubramanian, and S. Joshi. Robust statistics on riemannian manifolds via the geometric median. In *Proc. CVPR*, pages 1–8. IEEE, 2008.

[Gal11] J. Gallier. *Geometric methods and applications: for computer science and engineering*, volume 38. Springer, 2011.

[GB11] A. Grubb and J.A. Bagnell. Generalized Boosting Algorithms for Convex Optimization. *Arxiv arXiv:1105.2054*, 1:1–15, 2011.

[GBT] D. Gray, S. Brennan, and H. Tao. VIPeR: Viewpoint Invariant Pedestrian Recognition Dataset. `http://vision.soe.ucsc.edu/?q=node/178`.

[GBT07] D. Gray, S. Brennan, and H. Tao. Evaluating Appearance Models for Recongnition, Reacquisition and Tracking. In *Proc. PETS*, 2007.

[GHL04] S. Gallot, D. Hulin, and J. Lafontaine. *Riemannian geometry*. Springer Verlag, 2004.

[GL09] J. Gall and V. Lempitsky. Class-specific hough forests for object detection. In *Proc. CVPR*. IEEE, 2009.

[GMC+07] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. The CMU multi-pose, illumination, and expression (Multi-PIE) face database. Technical report, Carnegie Mellon University Robotics Institute. TR-07-08, 2007.

[Gou] N. Gourier. Head Pose Image Database (Pointing'04 ICPR Workshop). `http://www-prima.imag.fr/Pointing04/data-face.html`.

[GVL96] G.H. Golub and C.F. Van Loan. *Matrix computations*, volume 3. Johns Hopkins Univ Pr, 1996.

[GYR+11] J. Gall, A. Yao, N. Razavi, L. Van Gool, and V. Lempitsky. Hough forests for object detection, tracking, and action recognition. *IEEE Trans. PAMI*, 33(11):2188–2202, 2011.

[HALL05] C. Huang, H. Ai, Y. Li, and S. Lao. Vector Boosting for Rotation Invariant Multi-View Face Detection. In *Proc. ICCV*, pages 446–453, 2005.

[Hat02] A. Hatcher. *Algebraic topology*. Cambridge University Press, 2002.

[HJB+08] H. Hung, D.B. Jayagopi, S. Ba, J.M. Odobez, and D. Gatica-Perez. Investigating automatic dominance estimation in groups from visual attention and speaking activity. In *Proc. ICMI*, pages 233–236. ACM, 2008.

[HKR93] D.P. Huttenlocher, G.A. Klanderman, and W.J. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Trans. PAMI*, 15(9):850–863, 1993.

[HL08] J. Hamm and D.D. Lee. Grassmann discriminant analysis: a unifying view on subspace-based learning. In *Proc. ICML*, pages 376–383. ACM, 2008.

[HL09] J. Hamm and D.D. Lee. Extended Grassmann kernels for subspace-based learning. In *Proc. NIPS*, pages 601–608, 2009.

[HSDlTB11] D. Huang, M. Storer, F. De la Torre, and H. Bischof. Supervised Local Subspace Learning for Continuous Head Pose Estimation. In *Proc. CVPR*, 2011.

[HTF11] Trevor. Hastie, Robert. Tibshirani, and JH (Jerome H.) Friedman. *The elements of statistical learning (second edition)*. Springer, 2011.

[Jai] A. Jaiantilal. Random forests (matlab). `http://code.google.com/p/randomforest-matlab/`.

[JKF01] O. Jesorsky, K. Kirchberg, and R. Frischholz. Robust face detection using the hausdorff distance. In *Proc. AVBPA*, pages 90–95. Springer, 2001.

[JPC+09] N. Jojic, A. Perina, M. Cristani, V. Murino, and B. Frey. Stel component analysis: Modeling spatial correlations in image class structure. In *Proc. CVPR*, pages 2044–2051. IEEE, 2009.

[JWVG03] B. Jabarin, J. Wu, R. Vertegaal, and L. Grigorov. Establishing remote conversations through eye contact with physical awareness proxies. In *Proc. CHI*, 2003.

[Kar77] H. Karcher. Riemannian center of mass and mollifier smoothing. *Comm. Pure and Applied Math.*, 30(5):509–541, 1977.

[Kel75] J.L. Kelley. *General topology*. Springer Verlag, 1975.

[KH06] A. Krizhevsky and GE Hinton. *Learning multiple layers of features from tiny images*. PhD thesis, University of Toronto, 2006.

[KSB] T.K. Kim, J. Shotton, and S. Bjorn. Boosting & randomized forests for visual recognition. `http://www.iis.ee.ic.ac.uk/~tkkim/iccv09_tutorial`.

[Lan06] O. Lanz. Approximate bayesian multibody tracking. *IEEE Trans. PAMI*, 28:1436–1449, 2006.

[LBC+09] O. Lanz, R. Brunelli, P. Ian Chippendale, M. Voit, and R. Stiefelhagen. *Extracting Interaction Cues: Focus of Attention, Body Pose,*

and Gestures, pages 87–93. Number Human-Computer Interaction Series. Springer, 2009.

[LBK10] Y.M. Lui, J.R. Beveridge, and M. Kirby. Action classification on product manifolds. In *Proc. CVPR*, pages 833–839. IEEE, 2010.

[LCSL07] I. Laptev, B. Caputo, C. Schuldt, and T. Lindeberg. Local velocity-adapted motion events for spatio-temporal recognition. *CMUI*, 108(3):207–229, 2007.

[LD08a] A. Lablack and C. Djeraba. Analysis of human behaviour in front of a target scene. In *Proc. ICPR*, pages 1–4. IEEE, 2008.

[LD08b] Z. Lin and L.S. Davis. A pose-invariant descriptor for human detection and segmentation. In *Proc. ECCV*, pages 423–436. Springer, 2008.

[LKTT07] X. Liu, N. Krahnstoever, Y. Ting, and P. Tu. What are customers looking at? In *Proc. AVSS*, pages 405–410, 2007.

[LL05] Y.Y. Lin and T.L. Liu. Robust face detection with multi-class boosting. In *Proc. CVPR*, volume 1, 2005.

[LLF05] V. Lepetit, P. Lagger, and P. Fua. Randomized trees for real-time keypoint recognition. In *Proc. CVPR*, volume 2, pages 775–781. IEEE, 2005.

[LSB10] C. Leistner, A. Saffari, and H. Bischof. MIForests : Multiple-Instance Learning with Randomized Trees. In *Proc. ECCV*, pages 29–42, 2010.

[LSP06] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. CVPR*, volume 2, pages 2169–2178, 2006.

[LSS05] B. Leibe, E. Seemann, and B. Schiele. Pedestrian Detection in Crowded Scenes. In *Proc. CVPR*, pages 878–885, 2005.

[LSXFF10] L.J. Li, H. Su, EP Xing, and L. Fei-Fei. Object Bank: A High-Level Image Representation for Scene Classification & Semantic Feature Sparsification. *ML*, 1:1–9, 2010.

[LWB00] S.H.R Langton, R.J. Watt, and V. Bruce. Do the eyes have it? Cues to the direction of social attention. *Trends in Cognitive Neuroscience*, 4(2):50–58, 2000.

[LYT06] D. Lin, S. Yan, and X. Tang. Pursuing informative projection on grassmann manifold. In *Proc. CVPR*, volume 2, pages 1727–1734. IEEE, 2006.

[LZ04] S.Z. Li and Z.Q. Zhang. Floatboost learning and statistical face detection. *IEEE Trans. PAMI*, 26(9):1112–1123, 2004.

[LZHT08] M. Li, Z. Zhang, K. Huang, and T. Tan. Estimating the number of people in crowded scenes by MID based foreground segmentation and head-shoulder detection. In *Proc. ICPR*, pages 1–4, 2008.

[LZZ+02] S.Z. Li, L. Zhu, Z. Zhang, A. Blake, H. Zhang, and H. Shum. Statistical Learning of Multi-view Face Detection. In *Proc. ECCV*, pages 67–81, 2002.

[MBB00] L. Mason, P.L. Bartlett, and J. Baxter. Improved generalization through explicit optimization of margins. *ML*, 38(3):243–255, 2000.

[MBLS01] J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and texture analysis for image segmentation. *IJCV*, 43(1):7–27, 2001.

[MBM08] S. Maji, A.C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *Proc. CVPR*, pages 1–8. IEEE, 2008.

[MCT09] E. Murphy-Chutorian and M.M. Trivedi. Head pose estimation in computer vision: A survey. *IEEE Trans. PAMI*, 31(4):607–626, 2009.

[MG06] S. Munder and D.M. Gavrila. An experimental study on pedestrian classification. *IEEE Trans. PAMI*, 28:1863–1868, 2006.

[MJD+00] S.J. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler. Tracking groups of people. *CVUI*, 80(1):42–56, 2000.

[MNJ08] F. Moosmann, E. Nowak, and F. Jurie. Randomized clustering forests for image classification. *IEEE Trans. PAMI*, 30(9):1632–1646, 2008.

[MOZ02] Y. Matsumoto, T. Ogasawara, and A. Zelinsky. Behavior Recognition Based on Head-Pose and Gaze Direction Measurement. In *Proc. IROS*, volume 4, pages 2127–2132, 2002.

[MR03] R. Meir and G. Ratsch. An introduction to boosting and leveraging. *LNCS*, 2600:118–183, 2003.

[MS85] M. McKenna and R. Seidel. Finding the optimal shadows of a convex polytope. In *Proc. SoCG*, pages 24–28. ACM, 1985.

[MS11] I. Mukherjee and R.E. Schapire. A theory of multiclass boosting. *Arxiv arXiv:1108.2989*, 1:1–62, 2011.

[MSD97] A.W. Moore, J. Schneider, and K. Deng. E cient locally weighted polynomial regression predictions. In *Proc. ICML*. Citeseer, 1997.

[MSZ04] K. Mikolajczyk, C. Schmid, and A. Zisserman. Human Detection Based on a Probabilistic Assembly of Robust Part Detectors. In *Proc. ECCV*, pages Vol I: 69–82, 2004.

[MTF+10] S. Martelli, D. Tosato, M. Farenzena, M. Cristani, and V. Murino. An fpga-based classification architecture on riemannian manifolds. In *Proc. IMPRESS*, 2010.

[MTJ06] F. Moosmann, B. Triggs, and F. Jurie. Fast Discriminative Visual Codebooks using Randomized Clustering Forests. In *Proc. NIPS*, nov 2006.

[MYL+08] Y. Mu, S. Yan, Y. Liu, T. Huang, and B. Zhou. Discriminative local binary patterns for human detection in personal album. In *Proc. CVPR*, pages 1–8. IEEE, 2008.

[Ng07] A. Ng. Support Vector Machines. `http://cs229.stanford.edu/notes/cs229-notes3.pdf`, 2007.

[OB07] J.M. Odobez and S. Ba. A cognitive and unsupervised map adaptation approach to the recognition of the focus of attention from head pose. In *Proc. ICME*, pages 1379–1382. IEEE, 2007.

[Odo] J. M. Odobez. IDIAP head pose database. `http://www.idiap.ch/dataset/headpose`.

[Off08] UK Home Office. i-LIDS multiple camera tracking scenario definition. `http://tna.europarchive.org/20100413151426/scienceandresearch.homeoffice.gov.uk/hosdb/cctv-imaging-technology/i-lids/index.html`, 2008.

[OGX09] J. Orozco, S. Gong, and T. Xiang. Head pose classification in Crowded Scenes. In *Proc. BMVC*, 2009.

[OPM02] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. PAMI*, 24(7):971–987, 2002.

[OYTM06] K. Otsuka, J. Yamato, Y. Takemae, and H. Murase. Quantifying interpersonal influence in face-to-face conversations based on visual attention patterns. In *Proc. CHI*, pages 1175–1180, New York, NY, USA, 2006. ACM.

[Pen00] A. Pentland. Looking at people: Sensing for ubiquitous and wearable computing. *IEEE Trans. PAMI*, 22(1):107–119, 2000.

[Pen04] X. Pennec. Probabilities and statistics on Riemannian manifolds: a geometric approach. Technical report, INRIA, 2004.

[Pen07] A. Pentland. Social Signal Processing. *IEEE Signal Processing Magazine*, 24(4):108–111, July 2007.

[pet] Pets 2007. `http://pets2007.net/`.

[PFA06] X. Pennec, P. Fillard, and N. Ayache. A riemannian framework for tensor computing. *IJCV*, 66(1):41–66, 2006.

[Pic00] R.W. Picard. *Affective computing*. The MIT Press, 2000.

[PISZ10] J. Philbin, M. Isard, J. Sivic, and A. Zisserman. Descriptor learning for efficient retrieval. In *Proc. ECCV*, pages 677–691. Springer, 2010.

[Por05] F. Porikli. Integral histogram: A fast way to extract histograms in cartesian spaces. In *Proc. CVPR*, volume 1, pages 829–836. IEEE, 2005.

[PP08] K.B. Petersen and M.S. Pedersen. The Matrix Cookbook. *Technical University of Denmark*, 1:7–15, 2008.

[PPN09] M. Pantic, A. Pentland, and A. Nijholt. Special issue on human computing. *IEEE Trans. SMC*, 39(1):3–6, 2009.

[Pri12] S.J.D. Prince. *Computer vision: models, learning, and inference*, volume 67. Cambridge University Press, 2012.

[PSZ08] S. Paisitkriangkrai, C.H. Shen, and J. Zhang. Performance evaluation of local features in human classification and detection. *IET-CV*, 2(4):236–246, 2008.

[PT10] N. Payet and S. Todorovi. $(\mathrm{RF})^2$ - random forest random field. In *Proc. NIPS*, 2010.

[PZ79] J. Panero and M. Zelnik. *Human Dimension and Interior Space : A Source Book of Design*. Whitney Library of Design, 1979.

[ROM01] G. Ratsch, T. Onoda, and K.R. Muller. Soft margins for AdaBoost. *ML*, 42(3):287–320, 2001.

[RR06] N. Robertson and I. Reid. Estimating gaze direction from low-resolution faces in video. In *Proc. ECCV*, pages 402–415. Springer, 2006.

[RR11] N.M. Robertson and I.D. Reid. Automatic reasoning about causal events in surveillance video. *EURASIP Journal on Image and Video Processing*, 2011(1):530325, 2011.

[RTMF08] B.C. Russell, A. Torralba, K.P. Murphy, and W.T. Freeman. LabelMe: a database and web-based tool for image annotation. *IJCV*, 77(1):157–173, 2008.

[sam] Samurai project. `http://www.samurai-eu.org/`.

[SBB+06] R. Stiefelhagen, K. Bernardin, R. Bowers, J. Garofolo, D. Mostefa, and P. Soundararajan. The clear 2006 evaluation. In *Proc. CLEAR*, pages 1–44. Springer-Verlag, 2006.

[SBB+09] R. Stiefelhagen, K. Bernardin, R. Bowers, R. Rose, M. Michel, and J. Garofolo. The CLEAR 2007 evaluation. *Multimodal Technologies for Perception of Humans*, 4625/2008:3–34, 2009.

[SBOGP08] K. Smith, S. Ba, J. Odobez, and D. Gatica-Perez. Tracking the visual focus of attention for a varying number of wandering people. *IEEE Trans. PAMI*, 30(7):1–18, 2008.

[Sch] W. R. Schwartz. ETHZ dataset for appearance-based modeling. `http://www.liv.ic.unicamp.br/~wschwartz/datasets.html`.

[Sch02] Robert E. Schapire. The Boosting Approach to Machine Learning: An Overview. In *Proc. MSRI Workshop on Nonlinear Estimation and Classification*, 2002.

[SD09] W.R. Schwartz and L.S. Davis. Learning Discriminative Appearance-Based Models Using Partial Least Squares. In *Proc. SIBGRAPI*, 2009.

[Ser93] E. Sernesi. *Linear algebra: a geometric approach*. Chapman & Hall/CRC, 1993.

[Ser04] R.A. Servedio. Smooth boosting and learning with malicious noise. *JMLR*, 4(4):633–648, 2004.

[SFC+11] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *Proc. CVPR*, volume 2, page 3, 2011.

[SFYW99] R. Stiefelhagen, M. Finke, J. Yang, and A. Waibel. From gaze to focus of attention. In *Proc. VISUAL*, pages 761–768, 1999.

[SH11] C. Shen and Z. Hao. A direct formulation for totally-corrective multi-class boosting. In *Proc. CVPR*, pages 2585–2592. IEEE, 2011.

[SJC08] Jamie Shotton, Matthew Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *Proc. CVPR*, pages 1–8. IEEE, 2008.

[SKHD09] W.R. Schwartz, A. Kembhavi, D. Harwood, and L.S. Davis. Human Detection Using Partial Least Squares Analysis. In *Proc. ICCV*, 2009.

[SKP99] D.G. Sim, O.K. Kwon, and R.H. Park. Object matching algorithms using robust Hausdorff distance measures. *IEEE Trans. IP*, 8(3):425–429, 1999.

[SLHN10] S. Sommer, F. Lauze, S. Hauberg, and M. Nielsen. Manifold valued statistics, exact principal geodesic analysis and the effect of linear approximations. In *Proc. ECCV*, pages 43–56. Springer, 2010.

[SM07] P. Sabzmeydani and G. Mori. Detecting pedestrians by learning shapelet features. In *Proc. CVPR*, pages 1–8, 2007.

[SM09] R. Subbarao and P. Meer. Nonlinear mean shift over riemannian manifolds. *IJCV*, 84(1):1–20, 2009.

[SMSV11] M.J. Saberian, H. Masnadi-Shirazi, and N. Vasconcelos. TaylorBoost: First and second-order boosting algorithms with explicit margin control. In *Proc. CVPR*, pages 2929–2934. IEEE, 2011.

[Spe] M. Spera. Elementi di topologia. `http://www.di.univr.it/?ent=ava&cs=108&id=147&lang=en`.

[SS99] R.E. Schapire and Y. Singer. Improved Boosting Algorithms Using Confidence-rated Predictions. *ML*, 37:297–336, 1999.

[SSdVL03] P. Smith, M. Shah, and N. da Vitoria Lobo. Determining driver visual attention with one camera. *IEEE Trans. ITS*, 4(4):205–218, 2003.

[SWSW11] Chunhua Shen, Peng Wang, Fumin Shen, and Hanzi Wang. UBoost: Boosting with the Universum. *IEEE Trans. PAMI*, 1:1–8, 2011.

[SYW02] R. Stiefelhagen, J. Yang, and A. Waibel. Modeling focus of attention for meeting indexing based on multiple cues. *IEEE Trans. NN*, 13:928–938, 2002.

[SZ11] P. Sun and J. Zhou. AOSO-LogitBoost: Adaptive One-Vs-One LogitBoost for Multi-Class Problem. *Arxiv arXiv:1110.3907*, 1:1–17, 2011.

[TF10] D. Tran and D. Forsyth. Improved Human Parsing with a Full Relational Model. In *Proc. ECCV*, pages 227–240. Springer, 2010.

[TFC+10] D. Tosato, M. Farenzena, M. Cristani, M. Spera, and V. Murino. Multi-class Classification on Riemannian Manifolds for Video Surveillance. In *Proc. ECCV*, pages 378–391. Springer, 2010.

[TMF07] A. Torralba, K. Murphy, and W.T. Freeman. Sharing Visual Features for Multiclass and Multiview Object Detection. *IEEE Trans. PAMI*, 29(5):854–869, 2007.

[Tosa] D. Tosato. ARCO (ARrray of COvariance matrices), Code and Datasets. `http://sites.google.com/site/diegotosato/ARCO`.

[Tosb] D. Tosato. GDet (Group Detection) Head Pose Dataset. `http://sites.google.com/site/diegotosato/gdet`.

[TPM06] O. Tuzel, F. Porikli, and P. Meer. Region covariance: A fast descriptor for detection and classification. In *Proc. ECCV*, pages 589–600. Springer, 2006.

[TPM08] O. Tuzel, F. Porikli, and P. Meer. Pedestrian detection via classification on riemannian manifolds. *IEEE Trans. PAMI*, 30(10):1713–1727, 2008.

[Tri04] B. Triggs. Detecting keypoints with stable position, orientation, and scale under illumination changes. *Proc. ECCV*, I:100–113, 2004.

[Tu05] Z. Tu. Probabilistic boosting-tree: Learning discriminative models for classification, recognition, and clustering. In *Proc. ICCV*, volume 2, 2005.

[TVC08] P. Turaga, A. Veeraraghavan, and R. Chellappa. Statistical analysis on Stiefel and Grassmann manifolds with applications in computer vision. In *Proc. CVPR*, pages 1–8. IEEE, 2008.

[Vap98] V. Vapnik. Statistical learning theory, 1998.

[VF08] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`, 2008.

[VJ01] P. Viola and M. Jones. Rapid Object Detection using a Boosted Cascade of Simple. In *Proc. CVPR*, 2001.

[VJ02] P. Viola and M. Jones. Robust real-time object detection. *IJCV*, 57(2):137–154, 2002.

[VJV03] M. Viola, Michael J. Jones, and Paul Viola. Fast Multi-view Face Detection. In *Proc. CVPR*, 2003.

[VPB09] A. Vinciarelli, M. Pantic, and H. Bourlard. Social Signal Processing: Survey of an emerging domain. *JIVC*, 27(12):1743–1759, 2009.

[VS08] M. Voit and R. Stiefelhagen. Deducing the visual focus of attention from head pose estimation in dynamic multi-view meeting scenarios. In *Proc. ICMI*, pages 173–180, New York, NY, USA, 2008. ACM.

[VZ10] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. In *Proc. CVPR*, pages 3539–3546. IEEE, 2010.

[WAHL04] B. Wu, H. Ai, C. Huang, and S. Lao. Fast Rotation Invariant Multi-View Face Detection Based on Real Adaboost. In *Proc. FGR*, pages 79–84, 2004.

[WFDJ94] S. Whittaker, D. Frohlich, and O. Daly-Jones. Informal workplace communication: what is it like and how might we support it? In *Proc. CHI*, page 208, 1994.

[WHY10] X. Wang, T.X. Han, and S. Yan. An HOG-LBP human detector with partial occlusion handling. In *Proc. ICCV*, pages 32–39. IEEE, 2010.

[WMSS10] S. Walk, N. Majer, K. Schindler, and B. Schiele. New features and insights for pedestrian detection. In *Proc. CVPR*, pages 1030–1037. IEEE, 2010.

[WN05] B. Wu and R. Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In *Proc. ICCV*, 2005.

[WN07] B. Wu and R. Nevatia. Cluster boosted tree classifier for multi-view, multi-pose object detection. In *Proc. ICCV*, pages 1–8, 2007.

[WN08] B. Wu and R. Nevatia. Optimizing discrimination-efficiency tradeoff in integrating heterogeneous local features for object detection. In *Proc. CVPR*, 2008.

[WN09] B. Wu and R. Nevatia. Detection and segmentation of multiple, partially occluded objects by grouping, merging, assigning part detection responses. *IJCV*, 82(2):185–204, 2009.

[WS08] C. Wojek and B. Schiele. A performance evaluation of single and multi-feature people detection. *PR*, I:82–91, 2008.

[WSB+03] A. Waibel, T. Schultz, M. Bett, M. Denecke, R. Malkin, I. Rogina, and R. Stiefelhagen. SMaRT: the Smart Meeting Room task at ISL. In *Proc. ICASSP*, pages 752–755, 2003.

[WSJ+11] L. Wang, M. Sugiyama, Z. Jing, C. Yang, Z.H. Zhou, and J. Feng. A Refined Margin Analysis for Boosting Algorithms via Equilibrium Margin. *JMLR*, 12:1835–1863, 2011.

[YCWE07] P. Yin, A. Criminisi, J. Winn, and M. Essa. Tree-based classifiers for bilayer video segmentation. In *Proc. CVPR*, pages 1–8. IEEE, 2007.

[YKA02] M.H. Yang, D.J. Kriegman, and N. Ahuja. Detecting Faces in Images: A Survey. *IEEE Trans. PAMI*, 24(1):34–58, January 2002.

[YO08] J. Yao and J.M. Odobez. Fast Human Detection from Videos Using Covariance Features. In *Proc. VS Workshop*, 2008.

[YTCC09] Yi-Ping Hung Yu-Ting Chen, Chu-Song Chen and Kuang-Yu Chang. Multi-Class Multi-Instance Boosting for Part-Based Human Detection. In *Proc. ICCV Workshops*, pages 1177–1184, 2009.

[ZGX09] W. Zheng, S. Gong, and T. Xiang. Associating Groups of People. In *Proc. BMVC*, 2009.

[ZLSB10] B. Zeisl, C. Leistner, A. Saffari, and H. Bischof. On-line semi-supervised multiple-instance boosting. In *Proc. CVPR*, pages 1879–1879. IEEE, 2010.

[ZLY10] W.S. Zheng, J.H. Lai, and P.C. Yuen. Penalized preimage learning in kernel principal component analysis. *IEEE Trans. NN*, 21(4):551–570, 2010.

[ZMG08] P. Zehnder, E.K. Meier, and LJV Gool. An efficient shared multi-class detection cascade. In *Proc. BMVC*, 2008.

[ZPG+06] S.K. Zhou, JH Park, B. Georgescu, C. Simopoulos, J. Otsuki, and D. Comaniciu. Image-based multiclass boosting and echocardiographic view classification. In *Proc. CVPR*, 2006.

[ZZMC07] J. Zhang, S. Zhou, L. McMillan, and D. Comaniciu. Joint real-time object detection and pose estimation using probabilistic boosting network. In *Proc. CVPR*, volume 8, 2007.